PERSONALNI
računari

77

DECEMBAR 1991.
CENA 120 D

WINDOWS 3.1

TEHNIKE PROGRAMIRANJA
DOS 5.0 ISPOD HAUBE

SOFTVER
CLARION
DODACI ZA LOTUS
NORTON UTILITIES 6.0

PLOTERI
JET PRO 360 PLUS

PROCESORI
i486SX vs i486DX



YU ISSN 0352-7271

MEGATEST OSAM RAČUNARA

PERSONALNI
računari 77

časopis za informatiku
i računarstvo
YU ISSN 0352-7221
izdaje BIGZ
decembar 1991 cena 120 din.

Izdaje i štampa
Beogradski izdavačko-grafički zavod
11000 Beograd
Bulevar Vojvode Mišića 17
Generalni direktor
Ilija Rapajić
Direktor Novinskog sektora
Zoran Nikodijević
Glavni i odgovorni urednik
Jovan Regasek
Zamenik gl. i odg. urednika
Esad Jakupović
Tehnički urednik / sekretar redakcije
Natalija Desančić
Urednik za hardver
Dejan V. Veselinović
Stručna redakcija
Vesna Ćosić (aktuelnosti), Nenad Balečanin (baze podataka), Miodrag Lovrić (radna tabela), Dejan Ristanović (softver), Jovan Skuljan (programiranje), Dejan Veselinović (hardver), Dragutin Vuković (mraže), Zoran Životić (softver)
Adresa redakcije
11000 Beograd
Bulevar vojvode Mišića 17/III
Telefoni
653-748 (redakcija)
650-528 (prodaja)
Agencija BIGZ-a
(011) 651-793, 853-565
Teleks
11855 BIGZ YU
Telefaks
(011) 648 140, 647-855
Pretplata za zemlju
Za jednu godinu 1060
Za šest meseci 540
(na žiro-račun D.P. BIGZ, 60802-603-23264)
Pretplata za inostranstvo
Za jednu godinu 51 USD, 80 DEM, 607 CHS, 27 GBP, 271 FRF
(na devizni račun D.P. BIGZ 60811-620-16101-820701-999-03377)
Zbog usporenog prenošenja uplata preko banke molimo pretplatnike da nam posle svake nove uplate odmah pošalju foto-kopiju uplatnice
Rukopisi se ne vraćaju

SEZAM
sistem za modemske komunikacije
(011) 648-699, 24 časa dnevno
Upravnik sistema
Jovan Regasek
Administratori na sistemu
Zoran Životić i Dejan Ristanović
Moderatori
Dejan Balinda (CIVILIZACIJA), Jovan Bulajić (PC USER), Aleksandar Damjanović (PC OS), Slobodan Kalezić (ORKA), Zoran Kehler (MOST), Vladimir Kostić (PC SOFT), Vladimir Krstonošić (VICEVI), Esad Jakupović (RAČUNARI), Dalibor Lanik (ATARI ST), Pavle Pakovic (PC SOFT), Dejan Ristanović (NOVOSTI I KOMUNIKACIJE), Bojan Tepavčević (FORUM), Dejan Veselinović (PC HARD) i Zoran Životić (SEZAM)
Sekretarica sistema
Natalija Desančić
Softver
Zoran Životić (glavne aplikacije) i Dejan Ristanović (eksterni programi)
Hardver
Server: COMPAQ DESKPRO 386/25, 3 MB RAM a tvrdi disk 300 MB (COMPAQ Minhen). Mreža: dva stanice COMPAQ 286N/12, 1 MB RAM-a (COMPAQ Minhen), tri stanice „Ultra" AT 12 MHz, D-LINK DE 100 LanSmart ("Ultra", Skopje). Modemi ViVa (Thelems Communications, Beograd)

SADRŽAJ



Ako vam se čini da smo ovog puta malo preterali što smo na naslovnu stranu umesto jednog računara stavili čak četiri, onda se pripremite za još veći "šok" u časopisu: mega test se čak devet mašina dve firme (Gama Electronics, Beograd i MOST, Beograd). Manekenka je, ipak, samo jedna (ali vredna) – Tamara Jovanović iz Beograda. Računarski grupni portret sa damom napravila je ekipa: Nenad Marjanović (fotografija), Radmila Ludosan (make up), Branko Petkovic (frizura) i Rene (garderoba).

Na osnovu Mišljenja Sekretarijata za informacije SR Srbije broj 413-01-48 od 4. 2. 1991. na ovaj časopis plaća se osnovni porez na promet po povlašćenoj tarifi od 3 odsto

ŠTA IMA NOVO

Štampači

QMS PostScript na 600 x 600



QMS-PS 815 MR je pun naziv novog QMS laserskog štampača koji pored uobičajenih 300 x 300 dpi omogućava i štampanje PostScript dokumeneta u punih 600 x 600 tačaka po inču. Može se koristiti na svim PC-kompatibilnim i Apple Macintosh računarima kao i na većini mini i mainframe računara. Deklarisana brzina štampanja iznosi 8 strana u minutu a prebacivanje između HP i PostScript interpretacije (samim tim i između standardne i visoke rezolucija) se vrši automatski.

Windows

ZA LJUDE A NE KOMPJUTERE

Microsoft je nedavno predstavio *Microsoft Solution Series* kolekciju sastavljenu od tri samostalna paketa. To su *Microsoft Works for Windows*, *Microsoft Money* i *Microsoft Publisher*, svi namenjeni za rad pod *Windows 3.0* i dizajnirani za početnike i poslovne ljude koji nemaju vremena da ulaze u finese velikih programskih paketa.

Microsoft Works for Windows predstavlja *Windows* verziju dobro poznatog integrisanog paketa koji sadrži sve esencijalne module za poslovne aplikacije – tu su tekst procesor, program za unakrsna proračunavanja, baza podataka i modul za crtanje i grafičko predstavljanje podataka. Mogućnost prenošenja i kombinovanje podataka među ovim modulima se podrazumeva, a nova je *WorksWizards* funkcija. Ove funkcija predstavlja skup interaktivnih makroa koje omogućavaju korisniku jednostavan izbor i kreiranje tipskih dokumenata (adresari, poštanske etikete, cirkularna pisma itd).

Microsoft Money je program namenjen, kao što mu i samo ime govori, organizaciji i evidenciji finansija za pojedince i tzv. mali biznis. Na raspolaganju je niz opcija za maksimalno pojednostavljenje unosa i obrade podataka, kao i pregled postojećih podataka. Nije nevažno napomenuti da je program prilagođen bankarskom i finansijskom sistemu Sjedinjenih Američkih Država.

Microsoft Publisher predstavlja baznu DTP aplikaciju dizajniranu prvenstveno za kreiranje profesionalnih biltena i poslovnih formulara. Ovaj program omogućava korisnicima sa malo ili čak i nimalo znanja u oblasti slonog izdavaštva kreiranje i slaganje kvalitetnih dokumenata, pozivnica, čestitki, brošura itd. Kao i kod ostalih *Solution Series* aplikacija, sve funkcije su prilagođene ciljnoj grupi u smislu jednostavnosti i preglednosti i većina se može obaviti mišem. U tom smislu postoje tri pomoćne grupe opcija – *PageWizards* (interaktivni modul za obrazovanje strana), *WordArt* (biblioteka fontova i rutina za obrazovanje efekata na fontovima) i *BorderArt* (biblioteka okvira).

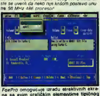
Microsoft Solution Series će se prodavati preko autorizovanih dilera *Microsoft Partners* po cenama od 395 DEM za *Microsoft Works for Windows* i *Microsoft Publisher* i 140 DEM za *Microsoft Money*. Svi programi rade na 286 ili jačim računarima pod *Windows 3.0*, sa 640 K RAM-a i 256 K extended RAM (za komforni rad se preporučuje puni megabajt), hard disk i MS kompatibilni miš.

Kontakt adresa: Microsoft GmbH, Eastern Europe Department, Edisonstrasse 1, 8044 Unterschleissheim, tel. 089/31705-453, fax 31705 598
Atlantis d.o.o., Cankarjeva 4, 61000 Ljubljana, tel./fax (061) 221-608

Baze podataka

LISICA NA MLAZNI POGON

Glavna odlika nove verzije (2.0) paketa za obradu podataka FoxPro, inače poznatog po velikoj brzini, je još veća brzina: na testovima PC Magazine-a *FoxPro 2.0* je postigao 9 (devet) puta brže rezultate od dBASE IV i šest puta brže od Paradox-a. Ovakve brzina je postignuta primenom nove tehnike za pristup podacima pod nazivom *Rushmore technology*. Efikasnost i brzina nove tehnike su tako očigledni da je oduševljeni urednik PC Magazine-a napisao doslovce: *FoxPro 2.0 je tako brz pri određenim operacijama da možete doći u iskušenje da zavirite pod poklopac vašeg AT-a kako biste se uverili da neko nije krišom postavio unutra 50 MHz 486 procesor*.



FoxPro omogućuje izradu efektivnih ekrana sa svim grafičkim elementima tipičnog Windows okruženja

Pored munjevite brzine, *FoxPro 2.0* može da se pohvali i određenim SQL setom komandi – reč je pre svega o komandi SELECT koja takođe koristi Rushmore tehniku kao deo svoje optimizacione strategije. Reč je o komandi koja se pokazala naročito efikasnom pri istovremenom pristupu podacima iz više različitih baza i koja je u stanju da zameni ceo niz standardnih komandi. Pored toga, tu su još dve SQL komande – CREATE TABLE i INSERT INTO – koje su potpuno integrisane u *FoxPro* jezik. Dalji dobitak na brzini i efikasnosti je postignut upotrebom novog tipa indeksiranja (CDX) i novog kompresionog algoritma koji automatski snima CDX fajl u sažetom formatu, što pruža mogućnost držanja većeg dela indeksnih datoteka u memoriji. Primenjeni kompresioni algoritam se pokazao veoma efikasnim – primera radi, smanjenje indeksne datoteke je toliko da dostiže samo jednu petinu iste datoteke kreirane u dBASE IV, uz 11% bolje vreme kreiranja indeksa.

Minimalna konfiguracija podrazumeva 512 K RAM-a, hard disk, DOS 2.0 ili viši, dok LAN verzije zahteva DOS 3.1 ili viši i odgovarajući mrežni softver koji podržava NetBIOS interfejs. Cena: $795 za standardnu i $1295 za LAN verziju.

Kontakt adresa: Fox Software Inc., 134 W. South Boundary, Perrysburg, OH 43551, tel. 419 874-0162

Komunikacije

KOD KUĆE NA POSLU

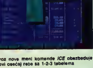
Sada možete da odete sa posla bez ijedne jedine diskete u džepu, čak i ne razmišljajući o tome šta sledeće treba da radite sve dok ne uključite svoj kućni PC. U tri sata ujutru. U pidžami. Tim rečima firma *Central Point Software* reklamira svoj novi paket namenjen preuzimanju kontrole nad računarom sa drugog računara. *Commute*, kako je novi paket nazvan, omogućava pristup svim podacima i programima koji se nalaze na udaljenom računaru preko modema ili Novell mreže. Moguće je pokretanje DOS ili *Windows* aplikacija, transfer fajlova, pristup e-mail-u, pa čak i štampanje na laserskom štampaču. Program raspolaže jednostavnim pull-down menijima, on-screen helpom i podržava miša. Kao takav je dobio i nedeljnu nagradu časopisa PC Week, a prodaje se po ceni od $79.

Kompajleri

ICE: JEFTINI 1-2-3 KOMPAJLER

Lotus 1-2-3 kompajleri pružaju mnoge prednosti za programere 1-2-3 aplikacija: pre svega kastomizaciju korisničkog interfejsa, ubrzanje rada, smanjenje potrebne memorije i prostora na disku, a uz sve to krajnji korisnici ne moraju da poseduju sam Lotus 1-2-3. Glavne mane ovakvih programa, pored nemogućnosti bilo kakvih izmena u jednom napravljenoj aplikaciji, je velika cena. Primera radi, *Baler XE* firme *Baler Software Corp.* se prodaje po ceni od $795. Ista firma je sada pustila na tržište kompromisni paket nazvan *Ice* koji po ceni od $395 i dalje pruža većinu prednosti skupljih kompajlera uz razliku u brzini i činjenici da krajnji korisnik mora da poseduje i sam Lotus 1-2-3.

Unakrsne tabele su „zamrznute" (ice – led) što znači da ne rade ništa brže od „običnih" tabela, ali zato mogu biti modifikovane uz mogućnost upotrebe većine (ali ipak ne svih) dodataka koji omogućavaju linkovanje „zamrznutih" tabela u standardne WK1 fajlove. Ice dodaje 70 novih ključnih reči (keywords) Lotusovom makro-jeziku i sve one služe uglavnom za prilagođavanje aplikacije specifičnim potrebama. Gotove aplikacije se, zajedno sa runtime modulom, zatim smešta na flopi disk pomoću posebnog *Build* programa. Ice zauzima 80 K konvencionalne memorije i zahteva prisustvo Lotus 1-2-3 v 2.2 ili 2.3.



Kroz nove meni komande *ICE* obezbeđuje novi osećaj rada sa 1-2-3 tabelama

Kontakt adresa: Baler Software Corp., 1400 Hicks Rd., Rolling Meadows, IL 60008, tel. 708 506-9700

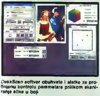
Softverske alatke

QEMM 6.0 – BETA VERZIJA

Menadžeri memorije za 386 mašine omogućavaju učitavanje drajvera i rezidentnih programa bez zauzimanja regularne memorije koja je inače potrebna za aplikacije. Uobičajeni način na koji se ovo postiže je upotreba slobodnog adresnog prostora između 640 K i 1 MB, koji se ne koristi za video svrhe, ROM-ove ili slične potrebe. QEMM 6.0 se razlikuje od ostalih rešenja po tome što prebacuje prostor rezervisan za ROM u EMS memoriju, a sam koristi upražnjeno mesto za učitavanje drajvera. Na ovaj način se dobija nešto manje mesta nego što su u stanju da obezbede konkurentski programi, ali u firmi *Quarterdeck Office Systems*, koja proizvodi QEMM, tvrde da je primenjeni metod mnogo čistiji.

Skeneri

IZLAZAK IZ SIVOG SVETA



***DeskScan* softver obuhvata i alatke za profinjenu kontrolu parametara prilikom skeniranja slike u boji**

Malo c u nazivu novog HP-ovog modela skenera označava njegove kolor mogućnosti, ali u njegovoj koncepciji nema ničeg malog izuzev cene. *HP ScanJet IIc* košta $2 195, odnosno isto onoliko koliko je monohromatski *ScanJet Plus* koštao prošlog leta (sada se prodaje po ceni od $1 590). Kao i njegov monohromatski prethodnik ScanJet IIc radi sa crno-belim i slikama sa 256 nijansi sivog. Pored toga, omogućava i skeniranje tačkastih i polutonskih kolor slika, kao i slika sa 256 boja i 24-bitnih kolor slika. HP opisuje 24-bitni mod kao „mod sa milionima boja". I mada ovo nije baš tehnički termin, ipak predstavlja prilično tačnu definiciju. Maksimalna mehanička rezolucija iznosi 400 dpi, ali priloženi softver omogućava postizanje rezolucije od 800 dpi.



Zahvaljujući 24-bitnom formatu, ScanJetIIc u skeniranju fotografije u boji obezbeđuje foto-realistički kvalitet

Korišćen je mehanizam sa jednom glavom i dve fluorescentne lampe za osvetljavanje, a ugrađen je i CCD (*charge coupled device*) sa tri trake koji omogućava kolor skaniranje u jednom prolazu. Podaci se u računar šalju preko priložene 8-bitne SCSI ekspanzione kartice, a u u pripremi je i Micro Channel verzija. Priloženi softver radi pod *Windows* okruženjem i celanja se prvenstveno na dva programa, takođe uključena u paket – *DeskScan* i *Publisher's Paintbrush*. *DeskScan* je vrlo sličan programu koji se isporučuje uz ScanJet Plus, ali je nešto pojačan i raspolaže dodatnim kolor opcijama.

Kontakt adresa: Hewlett Packard Co., 10310 Pruneridge Ave., Cupertino, CA 95014, tel. 800-752-0900

Stono izdavaštvo

VENTURA 4.0

Dugo očekivana 24-bitna kolor verzija paketa *Ventura Publisher 4.0 for Windows* se konačno pojavila na tržištu. Kompanija tvrdi da će nova verzija raspolagati funkcijama za kreiranje četvorobojnih fajlova za generisanje finalnog otiska na praktično svakom profesionalnom štamparskom sistemu. Verzija 4.0 podržava RGB, CMYK, CMY, HLS, FocalTone i Pantone modela za separaciju boja, a obezbeđena su i četiri nova DLL modula (DLL – *Dynamic Link Libraries*) za podešavanje boje, nazvana *Ventura Color Extensions*.

Ventura Scan je integrisani paket za skeniranje koji omogućava skeniranje direktno u Venturu.

Ventura Separator paket za separaciju boja na kompletnoj strani.

Ventura Photo Touch je program za obradu i retuširanje slika, i

Ventura ColorPro, specijalizovani alat za nezavisnu manipulaciju procesnim bojama.



***Ventura 4.0* podržava nekoliko standarda za 24-bitno razdvajanje boje, uključujući CMYK, RGB i Pantone.**

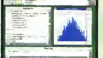
Pored toga, nova verzija raspolaže sa više novosti koje poboljšavaju interakciju sa *Windows* radnom sredinom, uključujući i podršku za DDE i OLE, a tu su i rečnik sa 88 000 reči, Undo/Redo i Search/Replace, filter za *Word for Windows* fajlove i prelom neregularnog teksta.

Nova verzija zahteva najmanje 386-baziranu mašinu sa 4 MB RAM-a i *Windows* 3.0, a poželjan je i matematički koprocesor. Osnovni paket se prodaje po ceni od $795, dok *Ventura Scan* košta $395, *Ventura Separator* $495, *Ventura Photo Touch* $795 a *Ventura ColorPro* „skromnih" $3 995.

Kontakt adresa: Ventura Software Inc. 15175 Innovation Dr., San Diego, CA 92128, tel. 800-822-8221, 619-673-0172

Bejzik

REALIZER VISUAL BASIC-a



Ovaj jednostavni program na *Realizer*-u u levom, dovoljan je za crtanje grafa u desnom prozoru

Realizer, kako je nazvana nova, moderna verzija bejzika namenjena programiranju *Windows* aplikacija, ruši zabludu da je za razvijanje *Windows* programa potrebno biti C programer. U cenu od $495 je uključena i kolekcija rutina za kreiranje i upravljanje prozorima, menijima i dijalozima, a proces programiranja teče vrlo jednostavno i direktno. *Realizer* se može porediti sa *Visual Basic*-om – iako je VB-ov pristup bliskiji načinu rada same *Windows* okoline. *Realizer*-ov proceduralni model se striktno drži modela strukturiranog programiranja i više odgovara korisnicima koji programiraju samo povremeno. Mada ne poseduje mogućnost rada sa višedimenzionalnim nizovima kao VB, *Realizer* nudi mnoge sofisticirane funkcije kojih u VB-u nema, na primer *Chart tool*, koji omogućava prikazivanje grafikona u raznim oblicima kao i razne formulare i tabele.

Realizer poseduje posebno jaku komunikacionu biblioteku, štampanje i animacionu biblioteku, a podržan je i pristup nezavisnim DLL-ovima i upotreba *Windows* DDE i *Clipboard*-a. U paket su uključena i dva pomoćna programa: *form editor*, sličan *Windows* SDK dialog box editoru i program za formiranje nezavisnih EXE fajlova. Oba ova programa, *FormDev* i *EXEbuilder*, su napisana u samom Realizeru.

Kontakt adresa: Within Technologies, Laurel Corporate Center, #201 South 8000 Midlantic Dr., Mt. Laurel, NJ 08054, tel. 609-273-8881

BRŽI PCL NA POSTSCRIPT PRINTERIMA

Kompanija *Laser Tools Corp.* je uspela da prevaziđe teškoće vezane za štampanje ne-Adobe PostScript aplikacija na PostScript štampačima i pustila na tržište program za automatsko prebacivanje. *Printer Control Panel* predstavlja rezidentni 5 K dug program koji menja tip emulacije (PostScript i PCL ili PostScript i ASCII) potpuno transparentno, zavisno od tipa štampača i željenog kvaliteta otiska.

Program, prvobitno zamišljen kao OEM dodatak za QMS laserske štampače i Adobe i Pacific Data kartridže, je takođe sposoban za kreiranje četiri virtuelna LPT porta, od kojih svaki raspolaže sopstvenim prekonfigurisanim tipom emulacije (za sada su podržani Diablo 630 i HPGL). Za rad je potreban Adobe PostScript štampač, 5 K RAM-a i DOS 2.1 ili viši. Cena: $149.

Kontakt adresa: LaserTools Corp., 1250 45th St., Ste 100, Emeryville, CA 94607, tel. 800-767-8004, 415-420-8777

Tvrdi diskovi

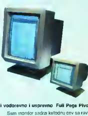
Najmanji hard na svetu

Firma *Integral Peripherals* će po svemu sudeći prve izneti na tržište najmanji do sada proizvedeni hard disk. Reč je o 1.82 uređaju sa početnim kapacitetom od 20 MB i prosečnim vremenom pristupa od 20 ms. Integral Peripherals 1820, kako mu glasi pun naziv, je, zahvaljujući svojim dimenzijama i niskoj potrošnji, idealan izbor za mobilne i ručne računare – firma je kao svoje ciljno tržište istakla *notebook* i *pen* PC je kao i uređaje za sakupljanje podataka. Uređaj je složen iz dva dela koji se mogu razdvojiti – jedan sadrži sam disk i glave i debeo je 10 mm, dok je elektronika smeštena u drugom 5.5 mm debelom delu. Firma predviđa pojavu 40 MB modela tokom sledeće godine.

Monitori

Okretni monitor

Korisniku paketa *Windows 3.0* nije potrebno dugo da shvati kako bajslensko šarenilo VGA rezolucija i boja ne znači mnogo na VGA monitoru standardne veličine (12-14"). Novi *Full Page Pivot* monitor u kombinaciji sa *SSGA MultiView* karticom omogućava prikazivanje pune strane u punom opsegu boja. Slično svom Mac uzoru, i ovaj 15" monitor se može okrenuti iz standardnog horizontalnog položaja (koji je izvanredan za tabelarne proračune i grafičke aplikacije) u vertikalni (prikazuje punu stranu teksta). Ova operacija se može izvesti samo jednom rukom, a senzor na bazi žive registruje rotaciju (samo punih 90 stepeni) i šalje signal grafičkoj kartici za odgovarajuću promenu orijentacije prikaza.



I vodoravno i uspravno: Full Page Pivot

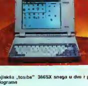
Sam monitor sadrži katodnu cev sa ravnom površinom (FST – Flat Square Tube) sa antirefleksionim slojem i zrnima veličine 0.28 mm. SSGA MultiView kartica je standardno opremljena punim megabajtom brzog video-RAM-a i raspolaže punim VGA spektrom rezolucija počev od 640 x 480 do 1024 x 768 piksela u 256 boja. Uz karticu se isporučuju i drajveri za *Windows 3.0*, *WordPerfect*, *Lotus 1-2-3*, AutoCAD, *AutoShade* i *Studio 3D*. Full Page Pivot je prvi potpuni kolor displej sistem sa prikazom pune strane za *Windows*, a za takvo zadovoljstvo treba izdvojiti $1698 uključujući i SSGA MultiView karticu. Za rad je potreban 386 ili jači računar na kome rade *Windows 3.0*.

Kontakt adresa: Radius Inc., 1710 Fortune Dr., San Jose, CA 95131, tel. 800-227-2795, 408-434-1010, fax 408-434-0770.

Računari

PRVOKLASNO JE LAKŠE

Već čuvenoj Toshibinoj portabl porodici se pridružio još jedan član – T2200SX predstavlja 20-MHz 386SX računar notebook klase koji zajedno sa baterijama teži tek nešto manje od 2.5 kg. Dimenzije od 29.8 x 21 x 3.8 cm predstavljaju redukciju od 33% u odnosu na model T2000SX, što je postignuto upotrebom novih nikl-hidridnih baterija, VGA kompatibilnog displeja (14.6 x 19 cm) i novog, tanjeg flopi diska. U računar se ugrađuju Conner hard diskovi od 40 ili 60 MB sa prosečnim vremenom pristupa od 19 ms. Dodatno smanjenje dimenzija je ostvareno upotrebom karbon-fiber polimera za izradu kućišta i ugradnjom nove šestoslojne matične ploče debljine od svega 5.8 mm.



Najlakša „tošiba": 386SX snaga u dva i po kilograma

Prelazak sa 20 na 10 MHz se ostvaruje pritiskom na funkcijski taster, a potpuno usporenje na samo 4 MHz sistem dostiže u sleep-modu, kada baterije mogu da traju i do 100 sati sa standardna 2 MB RAM-a. Za zahtevnije aplikacije i korisnike na raspolaganju je jedan slot za RAM-karte kojim se memorija može proširiti do 10 MB, slot za 2400 bps MNP 5 modem, serijski port, printer port koji može da se koristi i za eksterni 5.25" flopi i 150-pinski ekspanzioni port za Toshibin Desk Station IV. T2200SX košta $4499 sa 40 MB hard diskom, DOS-om i hipertekst dokumentacijom. Za 60 MB hard disk treba doplatiti još $300, a 2, 4- i 8-MB RAM kartice koštaju po $479, $889 i $1779 respektivno. 2400 bps modem sa MNP 5 protokolom košta dodatnih $359.

Kontakt adresa: Toshiba America Information Systems Inc., Computer Systems Division, 9740 Irvine Blvd., Irvine, CA 92718, tel. 800-334-3445, 714-583-3000, fax 714-584-6034.

Tastature

ZA MALE STOLOVE

Dve nove ultra kompaktne tastature *Mini KB* i *5100* engleske firme Electrone projektovane su specijalno za korisnike koji pate od nedostatka radnog prostora, ali tako da zadrže punu funkcionalnost.

Mini-KB je široka 33 cm, raspolaže sa 85 klik tastera, 12 funkcijskih tastera i kursorskim tasterima. Numerička tastatura je smeštena u zasebno kućište i može se priključiti direktno na Mini-KB, čime se postiže puni 101 raspored.

Model 5100 je još kompaktniji, nepunih 30 cm. Kompatibilnost sa tastaturama sa 102 tastera je postignuta dupliranjem funkcija pojedinih tastera. Obe tastature su XT i AT kompatibilne, a sa opcionim adapterom se mogu koristiti i na PS/2, a postoji i mogućnost naručivanja jezičkog rasporeda.

Cene: 79 Lstg za tastaturu, 29 Lstg za MiniPad (numerički dodatak).

Kontakt adresa: Electrone Ltd., Stonefield Way, Ruislip, Middlesex HA4 0QJ, U.K., tel. 44-81-864-5559, fax 44-81-864-2228.

Tvrdi diskovi

IZMENLJIVI HARD DISKOVI

Izmenljivi hard disk/kertridž drajv predstavlja kombinaciju flopi i hard diska: uz brzinu i kapacitet hard diska i izmenljivost medija flopi diska. SyQuest-ov *SyDOS 88* i *Transportable 90* firme Lomega predstavljaju najnovije modele na ovom delu tržišta.

SyDOS 88 je konfigurisan kao eksterni drajv, prodaje se zajedno sa Adaptec SCSI adapterom i DOS softverom, a izgleda i ponaša se kao standardni DOS hard disk. Kertridž se menja slično disketi – treba samo pritisnuti dugme na kućištu i sačekati da se vratašca otvore.

Model Transportable 90 koristi Bernoulli mehanizam (tehnologija koja postoji koliko i prvi PC) i 90-MB izmenljivu kasetu formata 5.25". Kasete se montiraju jednostavnim postavljanjem u vratašca, slično disketi, a drajv prima (samo za čitanje) i stare 44-MB kasete. Ovaj uređaj je dizajniran za prenošenje sa jedne mašine na drugu, a Lomega ga prodaje samo po porudžbini.

Cena: SyDOS 88, zajedno sa adapterom i jednim kertridžom od 88 MB, košta $1598, a za svaki dodatni kertridž treba platiti još $278.



I flopi i hard disk: SyDOS 88 i Transportable 90

Transportable 90 prodaje se za $1149, ali bez adaptera i kaseta. DOS Interface Kit (ISA) košta $265, MicroChannel $399, a Mac Interface Kit $49, dok set od tri kasete košta $687.

Kontakt adrese: SyDOS, 6501 Park of Commerce Blvd., Suite 110, Boca Raton, FL 33487, tel. 407-998-5400, fax 407-998-5414. Iomega Corp., 1821 West, 4000 South Roy, Utah 84067, tel. 801-778-1000, fax 801-778-3450.

Komunikacije

MicroChannel i X.25

Link 2501 kartica firme Netcomm Research omogućava povezivanje računara sa Micro Channel arhitekturom kao što su IBM PS/2 i RISC System/6000 na X.25 mreže. X.21bis (V.24/V.28) i X.21 (RS-422) interfejsi podržavaju X.25 brzine do 10 MB/s i postižu paketnu brzinu od preko 400 paketa u sekundi (128-bajtni podaci). Kartica koristi Motorola 68302 procesor koji je u stanju da rukuje paketima do 1 KB i raspolaže X.25 level 3 funkcijama

dok VLSI integrisana kola podržavaju rad sa X 25 level 2 funkcijama za glavnu vezu (sporedna veza se ostvaruje uz brzine prenosa do 250 000 bps) Glavni procesor podržava X 25 level 2 funkcije na sekundarnoj vezi, čime se smanjuje opterećenje hosta i ostavlja što je više moguće procesorske snage za glavnu aplikaciju

Uz Link 2501 se isporučuje softver za X 25(80) i X 25(84) aplikacije X 25(84) softver podržava CONS OSI (Connection Oriented Network Services) i prema tome aplikacije višeg nivoa kao što je X 400 Uz karticu se takođe isporučuje i XLPAD asembler/disasembler koji omogućava punu upotrebu X 25(84) mogućnosti i pruža niz terminalskih emulacija (uključujući i Videotex) uz maksimalno osam istovremeno aktivnih poziva pri čemu se za svaki poziv može koristiti različita emulacija Cena 995 Lstg

Kontakt adresa Netcomm Research Ltd Olympic Business Centre Paycocke Rd Basildon Essex SS14 3EX U K tel 44-268 534228 fax 44 268 534160

Clipper

GUI BIBLIOTEKA ZA CLIPPER

GForce je naziv nove *Clipper* biblioteke koja sadrži *low level* funkcije za piksel grafiku linije okvire uzorke itd kao i kompleksnije funkcije za kreiranje okruženja sličnog *Windows* sa 3-D dugmadima pull down menijima dijalozima ikonama fontovima bit-mapiranom grafikom i punom kontrolom miša sve to uz maksimalno 256 boja U paket su uključeni i font editor ikon editor i izvorni kod za oba programa Font editor je posebno zanimljiv budući da se sastoji od samo 400 linija u *Clipperu* pri čemu više od 100 linija predstavljaju opise dodeljivanja promenljivih Korisniku je omogućeno da ukloni bilo šta od toga upotrebom *Clipper* 5 hedera. Istovremeno je omogućeno korišćenje više fontova a ikone se mogu koristiti i za animacione efekte i logoe ili kao grafički meniji



***Clipper* kao *Windows*: Grafička biblioteka *GForce* omogućuje izgradnju modernih korisničkih interfejsa u *Clipper* aplikacijama**

GForce upravlja bitnom matricom preko *Clipperovih* stringova čime se omogućava pristup bit-mapiranim podacima iz samog *Clippera* Postojeće *Clipper* funkcije kao što je **strtran()** se mogu koristiti za promenu boja ili **substr()** za izvlačenje delova ili skaliranje bit-mape Pomoćni program za konverziju karaktera i stringova na relaciji PCX-Clipper omogućava konverziju i korišćenje PCX slika sa 16 ili 256 boja Biblioteka radi sa *Clipper* 5 i Summer 87 kompajlerima i linkerima sa dinamičkim maskama kao što je *Blinker*

Cena 185 Lstg

Kontakt adresa QBS Software Ltd 10 Barley Mow Passage Chiswick London W4 4PH U K tel 44 81 994 4842 fax 44-81-994 3441

KOMPAJLERI

PREGRŠT NOVIH PLUSEVA

Ljubitelji C++ -a su najavom tri nova Borlandova paketa došli na svoje – reč je o *Borland C++ 3.0*, *Borland C++ & Application Frameworks 3.0* i *Turbo C++ for Windows*; sva tri paketa sadrže C++ kompajler treće generacije.

Novi C++ 3.0 je profesionalni C i C++ kompajler sa zbirkom alata za kreiranje DOS i *Windows* aplikacija; novosti su:

- globalna optimizacija u cilju kreiranja manjeg i bržeg koda
- puna implementacija i ANSI C-a i C++ 2.1 sa maskama
- povećana brzina prevođenja C++
- mogućnost kompajliranja većih programa
- *ObjectBrowser* za vizuelno prikazivanje relacija među objektima i lakšu orijentaciju
- interfejs za rad pod DOS-om u zaštićenom modu sa odgovarajućom verzijom DOS kompajlera i pripadajućom radnom okolinom
- *Resource Workshop* za iscrtavanje *Windows* resurs objekata kao što su meniji, ikone dijalozi
- *EasyWin* biblioteka za konvertovanje DOS aplikacija za rad pod *Windows*
- prekompajlirana zaglavlja koja drastično smanjuju vreme potrebno za rekompajliranje

Pored toga *Borland C++* ne zahteva Microsoft-ov *Windows SDK* (*Software Development Kit*) a opremljen je i kompletnim setom alata za kreiranje DOS i *Windows* aplikacija u koji spadaju *Turbo Debugger* sa podrškom za optimizovani kod *Turbo Profiler* i objektno orijentisani *Turbo Assembler*

Borland C++ & Application Frameworks 3.0 predstavlja industrijski standard C++ uz dodatak *ObjectWindows* i *Turbo Vision* – paket uključuje novi C++ 3.0 i biblioteku za obrazovanje korisničkih interfejsa za *Windows* i DOS *Application Frameworks* sadrži biblioteku generičkih korisničkih interfejsa koji se mogu prilagođavati i proširivati objektnim programiranjem Biblioteka raspolaže velikim brojem gotovih objekata, uključujući menije dijaloge editore skrol-zone itd Uz obe aplikacije se dobija i kompletan izvorni kod a tu je i *Run Time* biblioteka u obliku izvornog koda

Turbo C++ for Windows predstavlja prvu *Windows* orijentisanu razvojnu okolinu za C i C++ i pruža pristupačno sredstvo za učenje *Windows* programiranja neprofesionalnim programerima U paket je uključen i *ObjectWindows* aplikacioni okvir za pojednostavljivanje *Windows*-a i grafički *ObjectBrowser* *Turbo C++ for Windows* je kompletno *Windows* programsko okruženje – za rad nije potreban *Windows SDK* Novosti koje ovaj paket donosi su

- Puna implementacija ANSI C a i C++ 2.1
- Integrisana *Windows* razvojna sredina
- *ObjectWindows*
- *ObjectBrowser*
- *SpeedBar* – zbirka prikladnih ikona za najčešće korišćene funkcije
- *Turbo Debugger for Windows*
- *Resource Workshop*
- *EasyWin* biblioteka

Verzije za početnike i neprofesionalne programere *Turbo C++* ($99 95) *Turbo C++ for Windows* ($149 95) i *Turbo C++ & Turbo Vision* ($199 95) Verzije za profesionalne programere *Borland C++* ($495 00) i *Borland C++ & AF* ($749 00) Doplata za registrovana korisnike se kreće od 89 95 (TC++ TC) i 99 95 (TP dBase) za Turbo odnosno 125 00 i 199 95 za Borland verzije (BC++ TC++Pro) Vlasnici programa BC++&AF 2.0 dobijaju novu verziju besplatno

TRŽIŠTE: KMA, BEOGRAD

KORAK PREMA KUPCU

Od pre nekoliko dana na našem tržištu mogu se nabaviti i računari poznate južnokorejske firme „Daewoo" (izgovara se kao „deevu" ali se može čuti i engleska varijanta „deivu") Bio je to jedan od povoda da posetimo distributera – malo preduzeće KMA iz Beograda

Lep ali nedovoljno veliki poslovni prostor firme KMA na drugoj galeriji Sava centra (dve prostorije) ovih dana je zatrpan kutijama sa računarima na kojima krupnim plavim piše „Daewoo" Firma trenutno pomalo podseća na magacin Ali neće dugo, kaže nam direktor Jelena Apostolović I to ne zato što je kutija svaki dan sve manje – znak da prodaja ide dobro – nego zbog toga što se KMA ovih dana preseljava sprat niže, u znatno veći prostor

Formirana maja 1989 godine kao komercijalno marketinška agencija, firma KMA, na čiju prvobitnu delatnost podseća još samo ime-akronim za svega dve i po godine stekla je ugled izuzetno sposobnog, pouzdanog i uspešnog partnera. Od marta 1990 KMA postaje d.o.o ovlašćeni distributer firme „Summagraphics" (doskora „Houston Instruments")

Visoko profesionalna prodaja plotera digitalizatora skenera, sekača i druge grafičke opreme ove poznate kuće zajedno sa odgovarajućim softverom i inženjeringom pružila je firmi KMA mogućnost da kompletira svoju ponudu kao i da formira mrežu dilera za prodaju proizvoda „Summagraphics" (Uzgred direktor KMA naglašava da izuzetno uspešnu prodaju velikim delom kako se pokazalo na osnovu anketiranja kupaca, mogu da zahvale kontinuiranom oglašavanju u „Računarima") O ukupnim efektima ovog posla najbolje svedoči činjenica da je KMA pre nekoliko meseci dobila najviše priznanje korporacije „Summagraphics" za zemlje Evrope i Bliskog istoka – za izuzetno uspešnu prodaju i podršku

Rukovodeći se, kako kažu u firmi „principom optimuma" KMA se odlučila za ravnotežu između (visokog) kvaliteta i (povoljne) cene Da su taj princip uspevali da održavaju i u najnepovoljnijim okolnostima mogu u velikoj meri da zahvale činjenici da je firma i izvoznik u neke susedne zemlje Sav uvoz u Jugoslaviju naime KMA pokriva sopstvenim izvozom Stotine korisnika grafičke opreme iz programa „Summagraphics" u Jugoslaviji nabavljene njihovim posredstvom najbolja potvrda su uspešnosti poslovne politike ove male firme U periodu mart prošle – mart ove godine na primer KMA je udesetostručila prodaju a njen direktor procenjuje da će se za period mart 1991 – kraj 1991 godine prodaja povećati još četiri do pet puta.

Zahvaljujući pažljivoj obradi tržišta i dobrim referencama KMA upravo ulazi u nove aranžmane za distribuciju opreme „Summagraphics" u nekim susednim zemljama ali – što je svakako novi kvalitet – uz prodaju odnosno izvoz znanja Ta sa svoje strane povoljno utiče na dalju ekspanziju firme U tom svetlu treba posmatrati i novi aranžman sa korporacijom „Daewoo" Reč je o džinovskoj južnokorejskoj firmi (čiji generalni zastupnik u Jugoslaviji je beogradski „Centrocoop") pozntoj u mnogim oblastima proizvodnje – od brodova preko hemikalija i automobila do računara.

Proizvodnjom računara bavi se kompanija „Daewoo Telecom Co" u sastavu korporacije Razumljivo je da se ovakva kompanija, koja sama drži osam odsto američkog tržišta računara, rukovodi dugoročnim planovima pa je i u slučaju saradnje sa KMA postavljena koncepcija ulaganja na duži rok. Zbog toga je KMA (odnosno novi deo firme koji zove KMAD) u mogućnosti da ponudi kvalitetne računare po privlačnim promotivnim cenama. Oprema je sklopljena u fabrici od komponenata poznatih svetskih proizvođača („Intel", „Texas Instruments" „Paradise" „Seagate") i atestirana – nije posredi, kako se kod nas kaže, „šrafciger" industrija.

Mašine južnokorejske firme moći će uskoro ne samo da se vide nego i isprobaju, jer KMA jednu od novih prostorija odvaja isključivo za stalne demonstracije opreme „Summagraphics" i „Daewoo" To je novi korak ove firme prema ozbiljnom kupcu koji ne želi samo da veruje nego i da se uveri

Esad Jakupović

PARADA VISOKE TEHNOLOGIJE

Ovogodišnji sajam SYSTEMS 91 u Minhenu je, kao i do sada, bio pun raznovrsnih novosti iz oblasti računarskog hardvera. Istovremeno održavanje čuvene Comdex/Fall manifestacije u Las Vegasu je, donekle, umanjilo prisustvo američkih firmi. Ovaj tekst je posvećen novitetima koji su se mogli „pohvatati" za vreme jednodnevnog „blic" obilaska.

Nebojša Novaković

Kao što je red, počećemo od svežeg PC hardvera. Dok se velika imena (Compaq, Acer, Mitac...) takmiče na polju višeprocesorskih 50 MHz 486 EISA servera sa brzim SCSI 2 sistemima paralelno vezanih diskova enormnih kapaciteta, sve nervoznije iščekujući novi 586 dotle i za obične smrtnike stižu lepe vesti. Pogledajmo, najpre, situaciju sa osnovnim pločama.

MATIČNE PLOČE

Kao što znamo, pojam „motherboards" u današnje vreme je neraskidivo vezan sa pojmom „Taiwan". Zato sam sebi dao slobodu da skočim do hale 23, gde je bila gomila štandova ljubaznih i za trgovinu uvek ornih Tajvanaca. Šta ima novo u najsnažnijoj kategoriji? Prvo loša: prva serija Intelovih 50 MHz 486 isporučenih tokom leta je pomerila početak masovnije prodaje ovih ploča na novembar. Zašto?

Naime, da bi omogućio pouzdan rad na 50 MHz bez hladnjaka i kojekakvih „IceCap" modula, Intel je morao da promeni i tehnologiju i unutrašnji raspored elemenata u 486. Tehnologija je sada 0.8 mikronska, a i funkcionalne celine 486 su malo preturbane. Na žalost, rezultat tog reda je imao ne malo bagova i uz to se suviše pregrevao i umirao posle svega nekoliko minuta rada. Smrt se mogla izbeći jedino uz dobar hladnjak ili IceCap, s kojim i 33 MHz 486 lepo radi na 50% višoj frekvenciji. Bagovi su do sada otklonjeni, termički problem takođe, pa bi nove 50 MHz 486 ploče, koje od ovog meseca kreću, trebalo da rade OK. Tajvanci su, u skladu s tim, na sajmu izložili nekoliko veoma zanimljivih 50 MHz 486 ploča po cenama tek 15-20% višim od cene 33 MHz prethodnika. Jedina mana: nijedna na sebi nije imala CPU.

50 MHz 486 generacija, bila ona ISA ili EISA, donosi sa sobom par tehničkih novosti. Prvo, 50 MHz procesor sa svojim 20 ns taktom i odgovarajućim zahtevima za brzinu u eksplozivnom 2-1-1-1 „burst" načinu rada na sabirnici, postaje prebrz za najveći deo današnjih 256-kilobitnih i 1-megabitnih SRAM čipova razumne cenovne kategorije koji se koriste za sekundarni keš. Da bi ploča bila maksimalno brza, tj. da je neka konkurentska ne bi tek tako prestigla, prvi i osnovni uslov je da komunikacija sa kešom i na ovim frekvencijama bude bez ikakvih stanja čekanja. U tom smislu navedeni 2-1-1-1 „burst" rad sa kešom znači da se prva 32 bita učitaju u 2 takt-ciklusa (40 ns na 50 MHz), a ostale tri 32-bitne reči, pošto se adrese za njih ne generišu na adresnoj sabirnici (podrazumeva se uzastopnost), svaka u po samo 20 ns! Ako od tih 20 ns oduzmemo vreme komunikacije CPU, sekundarnog keš kontrolera i SRAM, dobijamo da taj SRAM mora imati pristupno i ciklusno vreme od najviše 12 ns. Takvi 256-kilobitni čipovi postoje, ali ih još uvek proizvodi mali broj firmi i ono najgore, užasno su skupi. A problem su samo te tri uzastopne 32-bitne reči. One prva ne... Rešenje je nađeno u preplitanju dve banke SRAM (*two-way cache interleaving*). Po sistemu poznatom iz preplitanja DRAM, preplitanje keša omogućuje kontinualan 2-1-1-1 „burst" rad sa relativno lako nabavljivim 20 ns SRAM i time najbržu moguću komunikaciju sa kešom.



Druga novost je uvođenje „write-back" keša kao standard na praktično svim 50 MHz 486 čip setovima. Naime, komunikacija procesora sa memorijom se, pored čitanja podataka iz nje, sastoji i iz upisivanja nekih podataka u nju. Postoje tri tehnike rešavanja tog problema:

– „Write through", kao što i samo ime kaže, znači da procesor istovremeno upisuje podatak i u keš i u memoriju. Dobra strana je kontinualna identičnost ekvivalentnih podataka u kešu i u RAM, a loša strana je što prilikom svakog upisa (15 – 25% svih ciklusa) procesor mora da čeka na RAM (do 5 stanja čekanja na 50 MHz). Ovaj način je najjednostavniji i najsporiji i primenjen je kod Intelovog 82385 keš kontrolera za 80386.

– „Buffered write through" sadrži jedan ili više (do 4) 32-bitna bafera koji omogućuju da, kada procesor upisuje podatke u memoriju, oni budu istovremeno upisani u keš i u taj, jednako brz bafer, procesor nastavi rad bez čekanja na RAM, a prvog sledećeg takt-ciklusa keš kontroler sam prenese podatke iz bafera u RAM. Ovaj način je brži od prethodnog jer je većina upisa sa relativno malim brojem reči u jednoj turi, nešto složeniji je za izvođenje od prethodnog, a slabost su mu slučajevi sa dosta uzastopnih upisa, kada se bafer brzo napuni i opet dolazi do čekanja. Ovakvo rešenje je najrašireniје i naći ćete ga kod Chips&Tech PEAK/DM, ETEQ Cougar i Symphony Labs 386 i 486 čip setova.

– „Write-back" način je najnapredniji – sve podatke procesor upisuje samo u keš maksimalnom brzinom i odmah nastavlja rad, dok keš kontroler te podatke paralelno prebacuje u RAM. S ovim načinom se postižu najbolje performanse, ali je i njegovo izvođenje najsloženije. Imaju ga OPTI 386WB i 486WB i ETEQ Bengal ISA, kao i Symphony i TI EISA 486 čip setovi. Na sajmu smo tako mogli videti prve 50 MHz 486 EISA ploče sa do 1 Mb „write-back" keša i do 64 Mb RAM uz 7 EISA slotova, sve u *baby* formatu! Usled veoma velike konkurencije, od Nove Godine cene EISA ploča više neće biti bitno više od cena ISA ploča sa istim procesorom, tako da se EISA može širiti i van matenog tržišta PC file servera.

Takođe su se mogle videti i brojne konvertibilne 486SX/486DX ploče, gde prostom zamenom CPU i oscilatora od 20 MHz 486SX



Quantum Go-Drive formata 2.5 inča sa ugrađenim AT basom i SCSI kontrolerom donosi kapacitet od 60 odnosno 120 megabajta

dobijale 33 ili 50 MHz 486DX sistem. Neke od tih ploča su čak bile i u polovini *baby* formata pogodnoj za kompaktne mrežne stanice. 486SX je sada dostupan i na 25 MHz, kao i AMD 386SX. 286 ploče sa praktično više i ne izlažu.

TVRDI I MEKI DISKOVI

Diskovi svih vrsta, čiji je razvoj do nedavno daleko zaostajao za razvojem procesora i memorija, diže se sve bolje. Šta ima novo u svetu mekih diskova, koji će tokom iduće godine s nestankom 5.25 inčnih disketa sasvim izgubiti pravo na pridev „meki"?

Na sajmu su brojni nemački distributeri svih japanskih FDD proizvođača prikazali nova 2.88 Mb 3.5" flopi diskove. Odgovarajućih barijum ferritnih ED disketa ima, kao i tajvanskih kontrolera. MS DOS 5, DR DOS 6 i OS/2 2 ih podržavaju, a uz nova verzija čuvenog programa 800 ća na ove disketa moći da napakujete cela 3.3 Mb formatirano. Neki drugi 3.5" FDD, one firmi Brier i Insite na primer, mogu da na jednu disketu smeste 20 odnosno 40 Mb podataka zahvaljujući sistemu optički navođenog vertikalnog zapisivanja. Sa novim TDK metal disketama (setimo se metal audio kaseta) mogli bismo imati i 80 Mb formatirano. Međutim, pošto IBM obično postavlje standarde u ovoj oblasti, pogledajmo njegov predlog – 3.5" izbrisivi i neizbrisivi (WORM) magneto optički diskovi formatiranog kapaciteta 128 Mb. Drajvove rade Sony i Pinnacle Micro, brži su od ST 225 hard diska, a Verbatim i par drugih firmi već duže vreme nude obe vrste disketa, koje se inače ravnopravno mogu koristiti u istom drajvu. Ovi drajvovi su idealno rešenje za backup dvrdih diskova, pogotovo novih 3.5" laptopa.

Zaista, novi hard diskovi su prevazišli sva očekivanja, čak i moja, koja obično idu nebu pod oblake. Pogledajmo do kojih granica su stigli formatirani kapaciteti tvrdih diskova različitih formata.

2.5"	19mm visine	130 Mb
3.5"	25mm visine	525 Mb
3.5"	41mm visine	1050 Mb
5.25"	82mm visine	2650 Mb

Meni su bili najzanimljiviji novi 3.5" tvrdi diskovi od 1050 Mb formatirano, kao Quantum ProDrive 1050 i Seagate ST11200. Ti mali monstrumi ne samo što imaju za ovaj format donedavno nezamisliv kapacitet, nego ih odlikuje i stravična brzina – 10 ms prosečno pristupno vreme i do 10 megabajta u sekundi praktična brzina prenosa preko Fast SCSI-2 interfejsa. Ova brzina prenosa se postiže ne samo izuzetno velikom mehaničkom brzinom diska, nego i velikom (256 ili 512 K) inteligentnom keš memorijom na disku.

Primećuje se, takođe, da se stari ST506 i ESDI standardi polako povlače. Brzina, inteligencija i široka upotrebljivost SCSI-2 i IDE interfejsa jednostavno su predodređene da pobede. Uz to, na tržištu je sada veliki broj brzih i visokointeligentnih SCSI-2 procesora koji se neposredno mogu vezivati na 32 bitne 386/486



Ljuti otrov u malom pakovanju: Compaq Portable 486c sa kolor grafikom i EISA basom

sabirnice. Chips&Tech proizvodi i verziju za laptop i notebook računare, dok najnoviji NCR 53C720 Fast Wide SCSI 2 32 bitni procesor sa podiskom 8 bitnom, ali i novom 16-bitnom SCSI, postiže brzine od 20 megabajta/s na SCSI strani i celih 106 megabajta/s na 33 MHz 486 strani, radeći kao koprocesor sa svojim ultrabrzim DMA upravljačima. A ceo posao se rešava u jednom čipu. Pošto su ovakvi upravljači brži od bilo koje PC sabirnice (uključujući i EISA), a sve je u jednom komadu, logično rešenje je smeštanje na osnovnu ploču. Na taj način komunikacija se svim hard diskovima, strimerima, optičkim diskovima i ostalim SCSI periferijama ostvaruje se preko inteligentnog kontrolera na samoj matičnoj ploči. Slično brzo rešenje za Ethernet i Token Ring mreže Intel prodaje pod nazivom 82596CA.

GRAFIKA

VESA standard je konačno u potpunosti prihvaćen, praktično primenjen (nova generacija ET4000 kartica) i sve bolje softverski podržan (*Autodesk Animator Pro*) na polju Super-VGA grafike. Nove kartice sa VESA video BIOS-om omogućavaju rezolucije 1280×1024 u 16 boja sa preplitanjem, 1024×768 u 256 boja na ergonomskih 70 Hz i 800×600 u 32768 boja za aplikacije tipa *AutoShade* ili *3D Studio* kojima pogoduje takav izodr odja. Softverska podrška u vidu hrpe drajvera uz svaku karticu je sva odlja, pogotovo na polju CAD.

Ako se popnemo jednu stepenicu gore, vidimo da žestoka bitka između 8514A (XGA) i TIGA standarda zapravo tek počinje. Novi 8514A i (delimično) XGA-kompatibilni Chips&Tech 82C480 procesor po hardverskoj brzini crtanja linija pokazuje gotovo jednake performanse kao i Texas 34020, uz naravno znatno nižu cenu finalnog proizvoda i mogućnost integracije na samu matičnu ploču. Na štandu Texas Instrumentsa pokušali smo da doznamo da li će već iri godine stari 34020 – 34082 par, koji je ipak zasnovan na tehnologiji kojom su rađeni MC 68030 i Intel 80386, uskoro dobiti naslednika koji bi se mogao takmičiti sa konkurencijom, pogotovo Intelovom 860 porodicom. Odgovor nismo uspeli da dobijemo. U 8514A/XGA porodici najzanimljivije su kartice ATI 8514/Ultra i Matrox Magnum, a u TIGA porodici (low end) Hercules Graphic Station +2 i NDI Volante AT1000.

Na najvišoj stepenici PC grafike, 3-D grafičkim karticama, čini se da Intelova 860 porodica odnosi pobedu kao standard. Nove 860-bazirane 3-D podsisteme za PC predstavili su Hercules (Superstation 3D), SPEA (FGA860), Datapath (Merlin), Kontron i još par drugih firmi. Osnovna karakteristika ovih sistema je višestruko (i do 10 puta) ubrzanje 3-D grafičkih zadataka, od editovanja crteža do kvalitetnih senčenja i animacije, u odnosu na najbrže današnje 486 PC. Sve ove kartice, pored 860 dela sa do 16 Mb RAM, sadrže i jedan od Texasovih grafičkih procesora za 2-D ekranska operacije i TIGA kompatibilnost. Na raspolaganju su i posebni programski paketi napisani u 860 kodu, kao izvrsni *RenderStar 2* za ultrabrzo fotorealistično senčenje i animaciju. Od Nove godine na tržištu ća se naći nova generacija 860 kartica, koja će imati jedan ili više novih i860XP procesora, donoseći dodatno ubrzanje i do tri puta u odnosu na sadašnje 33 MHz 860 kartice. Tokom iduće godine, pošto se standardizacija 860 kompajlera i razvojne okoline dovršava, možemo očekivati još odlju softversku podršku ovom procesoru. Autodesk je već najavio verzije svih svojih paketa za 860.

Za rezolucija ovih kartica reda 1280×1024 u 32768 boja na primer, potrebni su i odgovarajući monitori. Prijatno me je iznenadila činjenica da, pored Japanaca, i drugi dalekoistočni proizvođači usvajaju kao obavezne švedske SSI norme za zaštitu od zračenja, kao i vertikalne frekvencija osvežavanja od 70 i više Hz. Samsung, TVI, Sampo i drugi su izložili upravo takve monitore dijagonala od 17 do 21 inča, sa takođe obaveznim ravnim ekranima.

Sa dolaskom 50 MHz 486, procesorska snaga personalaca izgleda ozbiljno i proizvođačima grafičkih radnih stanica. Sun je uvideo da 50 MHz 486 i njegov 40 MHz SPARC u goloj brzini stoje rame uz rame. Pored obaranja cena svojim SPARC radnim stanicama, Sun sada i uz najslabije 33 MHz mono SPARC modele daje grafički ubrzivač snage nešto veće od 34020. Inače, *Lotus 1-2-3* i *AutoCAD 11* izgledaju divno na UNIX prozorima Sparcstation 2.

Sve u svemu, na velikom sajmu tipa Systems za jedan dan se ne može baš sve obići, ali ovaj kratak rezime vam može ukazati na novosti koje su izlagači predstavili i na neke trendove koji se mogu očekivati tokom naredne godine.

ZABLUDE O 80486

80486 je među nama već dve godine (80386 dosta duže), a ipak se o njemu u suštini jako malo zna. Pominjemo ga, naravno, kao centar raznih moćnih PC platformi, hvalimo njegovu brzinu i fleksibilnost, „mučimo" ga benčmark testovima, ali ga u svim tim prilikama tretiramo kao „malo" brži 8086/8088. O njegovim pravim 32-bitnim mogućnostima zna se jako malo, a i ono što se „zna" često nije tačno. Ovaj napis posvećujemo nekim od tih zabluda.

Na sve tvrdnje koje ćemo u sledećim redovima „demantovati" smo negde naišli – većinu smo pročitali u domaćim ili stranim računarskim publikacijama, nešto se čulo u razgovorima (u kojima su tvrdnju svi prisutni prihvatali kao neospornu) u ponešto smo i sami odnekud bili ubeđeni. Pa ipak, čak i površan pregled dokumentacije mikroprocesora 80386 i 80486 pokazaće da se radi o zabludama. Napomenimo još da svaka „zabluda" ne mora uvek biti sasvim pogrešna – neke od stvari koje slede naprosto nisu dovoljno precizno iskazane, a ima i situacija da je tvrdnja gotovo tačna, ali ne iz razloga koji se navode (kao kada, rešavajući neki zadatak, pogrešite dva puta, greške se potru pa rezultat ispadne tačan). No da krenemo.

1. Mikroprocesori 80386 i 80486 su sa softverske strane identični. Zapravo, ne postoji način da program sazna da li se izvršava na 80386 ili na 80486.

Ovu frazu sam toliko puta ispisao da se prosto bojim da mi ovo nije prvi put. Nešto slično je pisalo u BYTE-u pre par godina, a pominje se i na BIX-u s vremena na vreme. Verovatno i na mnogim drugim mestima. A sa istinom nema mnogo veze – između 80386 i 80486 postoje itekako značajne softverske razlike, naravno u sferi uobičajenoj za Intelove procesore. 80486 u sebi „sadrži" kompletan 80386 (vertikalna kompatibilnost), ali ima i neke nove elemente. Čak tako velike kao što su registri. 80486, na primer, uvodi registre TR3, TR4 i TR5. Ova tri konkretna registra omogućavaju dijagnostičkom softveru (npr. kodu na početku BIOS-a) da proveri internu keš memoriju pristupajući TLB-u (*translation lookaside buffer*). Osim toga, 80486 ima i jedan novi fleg zvani AC (*alignment check*) čije postavljanje omogućava dobro pisanim operativnim sistemima (ima li takvih?) i kompajlerima da znatno ubrzaju izvršni kod time što će „zagarantovati" mikroprocesoru da svaki podatak koji će se čitati ili pisati počiva na adresi deljivoj sa 4. Ukoliko se docnije pokaže da nije tako, izvršavanje traje daleko duže nego što bi trajalo da fleg nije setovan. Ima i novih instrukcija specifičnih za 80486, premda su mnoge od njih po značaju marginalne. Da ne ostanemo bez dokaza i za ovu tvrdnju, pomenućemo instrukciju BSWAP.

2. Uz mikroprocesor 8086 koristi se aritmetički koprocesor 8087, uz 80286 – 80287, uz 80386 ide 80387 a uz 80486? Naravno, 80487.

Dobro, dobro, nisam ni tvrdio da neko od čitalaca ovog teksta stvarno tako misli. Pa ipak, tvrdnja je više puta izrečena (recimo na našoj televiziji) pa zaslužuje da se nađe na spisku. Integralni deo mikroprocesora 80486 je jedinica za operacije za brojevima u pokretnom zarezu koja je vertikalno kompatibilna sa prethodnim Intelovim aritmetičkim koprocesorima. Ili, prostije i nepreciznije rečeno, u 80486 je već „ugrađen" 80487.

Dejan Ristanović

Znatno posle 80486, pojavio se 80486SX u koji ovaj koprocesor nije „ugrađen". Ploče sa 80486SX u ovom trenutku predstavljaju prilično lošu kupovinu (ako vam, recimo za DTP, treba snaga 80386/80486 ali vam ne treba koprocesor, napravićete bolji posao ako uzmete 80386 na višoj frekvenciji), ali to je već druga priča.

3. Interni aritmetički koprocesor ugrađen u 80486 se ne može isključiti.

Ako je baš potrebno, može. Flegovima ET i EM u prvom kontrolnom registru (CR0) može se simulirati sistem bez koprocesora.

4. Ukoliko 80387 ne postoji, 80386 ga emulira, doduše uz logičan gubitak na performansama.

Na žalost, nije tako. 80386 ne može „tek tako" da emulira 80387. Ali, setovanjem EM bita u CR0 registru (*emulate math coprocessor*) procesoru se nalaže da, čim naleti na instrukciju za koprocesor, generiše interapt (*trap*) i preda kontrolu operativnom sistemu. Tamo može biti potprogram koji realizuje željene operacije ili možda komunicira sa nekim koprocesorom nezavisnog proizvođača. Neki 80386 BIOS-i zaista sadrže takve programe.

5. 80486 ima interni keš od 8K. Taj keš se ne može isključiti.

Ne samo da se može isključiti, nego se može kontrolisati da li će taj keš biti *read through write through* (za šta bi to služilo?) ili *read/write*. Za to su zaduženi flegovi CD i NW (*Cache disable* i *No write through*) u CR0.

6. Nepostojeća instrukcija se na 80486 tretira kao NOP (No Operation).

Kladio bih se da je ovu tvrdnju (pročitao sam je nedavno u jednom domaćem kompjuterskom časopisu) izrekao neki bivši vlasnik *Commodore 64* ili nekog drugog računara zasnovanog na mikroprocesoru 6502 firme MOS Technology. Ovaj mikroprocesor je, naime, „ludeo" kad naiđe na nepostojeću instrukciju, a onda su napravljeni 65C02 i neke druge varijante 6502 koje su nepostojeće instrukcije zaista tretirale kao NOP, što je bio priličan korak napred.

Na 80286/386/486 se otišlo još dalje – nailazak na nepostojeću instrukciju izaziva takozvani *invalid opcode fault*, tj. predaje kontrolu rutini koja opslužuje INT 6. Ista stvar se dogodi čak i kada je neki od operanada neispravan ili kada korisnik pokuša da izvrši neku 32-bitnu instrukciju dok se nalazi u virtualnom 8086 modu.

7. 80486 je, kada radi u realnom modu, sasvim kompatibilan sa 8086 i 80286. Svaki program koji je radio na njima radiće i na njemu.

Skoro da je tačno, ali ipak, reč **svaki** je malo prejaka! Kompatibilnost, na primer, „pada" baš na nepostojećim instrukcijama, i to naročito na relaciji 80286-80486. Ukratko rečeno, neki kod koji je na 8086 bio nepostojeća instrukcija i izazivao neko nedokumentovano ponašanje mikroprocesora, odnosno koji je na 80286 izazivao pomenuti *invalid opcode fault* na 80386/486 može da bude sasvim legalna instrukcija. Teško da bi neki programer na ovome zasnovao neki komercijalni proizvod (možda eventualno neku zaštitu?), pa je u praksi tvrdnja 7 „dovoljno" tačna. Ipak, teorijski bi mogao da nastupi „kuršlus".

Postoji i dvadesetak drugih razlika, ali su sve one jako periferne. 80486, na primer, ne može da „podnese" instrukciju dužu od 15 bajtova, dok 8086 (zahvaljujući daleko primitivnije rešenoj fazi pripreme naredbe) ne pravi pitanje oko dužine instrukcije. Instrukcija duža od 15 bajtova normalno ne postoji, ali se može sintetisati korišćenjem gomile redundantnih prefiksa.

8. 80286, 80386 i 80486 se ne mogu „vratiti" iz zaštićenog u realni mod. Jedini način da se to izvede je resetovanje mikroprocesora.

Uh! Ima li neko ko ne misli tako? Bez obzira na to koliko smo je puta ponovili (npr. u „Računarima"...), tvrdnja je tačna samo u slučaju 80286 – njegovi tvorci su sa očito „plašili" da će se neka aplikacija prebaciti iz zaštićenog u realni mod, a onda po volji vršljati memorijom. Na 80386 i 80486 je uvedena „policija" koja se o tome brine, pa se prostim setovanjem i resetovanjem PE (*protect enable*) flega u CR0 „šeta" između zaštićenog i realnog moda. Naravno, ukoliko proces ima privilegije da na taj fleg utiče.

9. 80386 i 80486 su prevazišli segmente – u zaštićenom modu memorija je linearna ali...

Ovo i nije toliko pogrešno koliko je neprecizno. 80386 i 80486 **uvek** rade sa segmentima.

10. ... kada se radi sa segmentima, onda je svaki segment ograničen na 64 K.

Srećom, nije! Segment je sada ograničen na 4 gigabajta (2^{32}), što je ujedno i adresni prostor svakog 32-bitnog procesora. Dakle, koliko god da imate memorije, možete je čitavu staviti u jedan segment, pa zbog toga i rekosmo da prethodna tvrdnja nije precizna ali je u suštini tačna. Segmente ćete, dakle, koristiti kada su zaista potrebni, a to i nije tako redak slučaj – višeprogramski rad je, na primer, lakše organizovati uz segmentiranu memoriju. Ukoliko se, sa druge strane, portira neki operativni sistem koji je zasnovan na linearnoj (*flat*) adresnoj mapi (npr. *Unix*), segmenti se lako zaobiđu.

11. Kada mikroprocesor u zaštićenom modu radi sa virtualnom memorijom, segment je ograničen na 1 megabajt.

Znajući prethodno, tvrdnja očito nije tačna. Pa ipak, nedavno su uspeli da me „ubede" da jeste. Dokaz „ide" otprilike ovako: kompletan rad sa virtualnom memorijom na 80386 i 80486 zasniva se na takozvanim deskriptorima segmenata – neka vrsta mape koja govori gde

se u realnom adresnom prostoru nalazi određena virtuelna stranica memorije. Prva reč (16 bita) deskriptora označava se kao *limit* – tu piše koliko je segment veliki kako aplikacija ne bi usled greške programera „bombardovala" tuđi adresni prostor. Tako je bilo na 80286, gde je segment bio ograničen na $2^{16}=64K$. Kod 80386 i 80486 polje *limit* je „dobilo" još i prva četiri bita četvrte reči deskriptora (na 80286 ta četvrta reč je obavezno bila 0), pa je dugačko ukupno 20 bita. 2^{20} je tačno 1 megabajt što je, dakle, i maksimalna dužina segmenta!

Zvuči ubedljivo, ali ipak nije tačno. U stvari, Intelovi inženjeri očito nisu imali prostora za čitavo polje *limit* pa su se snašli i uveli jedan fleg u poslednjoj reči deskriptora koji se zove G (*granularity*). Ako je taj fleg nula (a na 80286 je uvek nula jer čitava poslednja reč deskriptora mora da ima vrednost nula), *limit* je izražen u bajtovima. Ako je G=1, *limit* je izražen u stranicama, pri čemu je jedna stranica 4096 bajta. Kada pomnožite 2^{20} sa 4096 dobijate tačno 2^{32}, odnosno 4 gigabajta, što je maksimalna dužina segmenta na 80386/80486. Vešto smišljeno, zar ne?

12. Kada se stek prepuni, dešavaju se razne ružne stvari koje često rezultiraju potpunim padom sistema.

Svakako istinito na starim Z80, 6502, 8086... srećom tehnika napreduje. Pre svega, stek segment ima svoj deskriptor i svoj limit – čim se on pređe, proces se privremeno suspenduje i kontrola vraća operativnom sistemu koji može nešto da spasava ili da definitivno prekine program kome treba više steka nego što je predviđeno. Čitava provera obavlja se potpuno automatski i ne oduzima nikakvo vreme.

Prosta kontrola granica nije sve što 80486 nudi – uvedeni su i segmenti koji „rastu" zvani *expand down segments*. Za razliku od „normalnog" segmenta čije adrese počinju od 0 i rastu do polja *limit* pridruženog deskriptora, rastući segmenti počinju od *limit+1* i rastu do FFFFFFFFh. Ukoliko je u takvom segmentu stek i ukoliko nastupi prekoračenje, operativni sistem može jednostavno da umanji vrednost *limit*-a (naravno, ako ispod ima prostora) i dopusti programu da se dalje izvršava. Sve ovo, jasno, važi samo ako se program sve vreme izvršavao u zaštićenom modu – u realnom nas prosto sledi krah sistema!

13. U zaštićenom modu postoje dve vrste procesa – sistemski, kojima je „sve" dopušteno, i korisnički, koji podležu ograničenjima.

80386 i 80486 omogućavaju i više od toga! Zapravo, postoje čak četiri nivoa – najprivilegovaniji nosi broj 0 i trebalo bi da se radi o samom jezgru operativnog sistema, tj. o kodu kome je (da damo materijala za tekst „Zablude u tekstu o zabludama o 80486") sve dopušteno. Nivo 1 je „ostatak" operativnog sistema, nivo 2 su važni sistemski programi, a nivo 3 korisnički programi sa najmanjim pravima.

To je teorija. U praksi uglavnom nema potrebe za ovolikim „deteljisanjem". Razne implementacije Unix-a, na primer, koriste samo dva nivoa privilegija – 0 je kernel, a 3 korisnički programi. OS/2 (verzija 1.XX, još ne znamo kako stoje stvari na OS/2 2.0, ali najverovatnije nema izmena po tom pitanju) koristi tri nivoa – 0 je operativni sistem, 2 su programi koji smeju da pristupaju uređajima u sistemu (npr. serijskom portu itd), a 3 ostale aplikacije.

14. Programi koji se izvršavaju na višem prioritetnom nivou mogu da pristupaju podacima i pozivaju potprograme koji su na nižem prioritetnom nivou. I obrnuto – program na nižem prioritetu ne može da pristupa podacima niti da poziva potprograme višeg prioriteta.

Nešto slično nigde nisam pročitao, ali nije da nema logike, bar na prvi pogled. Na drugi pogled, stvar je prilično sumnjiva – kako će, na primer, program nižeg prioriteta da zatraži neku uslugu operativnog sistema ako ne sme da pozove operativni sistem? A opet, ako mu dopustimo da poziva operativni sistem po volji, kako da budemo sigurni da ga neće „gađati" u neko nepredviđeno mesto (koje uopšte i nije ulazna tačka) i tako u boljem slučaju srušiti sistem, a u gorem obezbediti sebi neke privilegije koje ne bi smeo da ima?

Da vidimo kakva je stvarna situacija. Izrečena tvrdnja važi što se tiče podataka – program iz jezgra operativnog sistema (prioritet 0) može da pristupa svemu, program prioriteta 1 može da pristupa i podacima koji pripadaju programu prioriteta 2 i 3 i tako dalje. Nasuprot tome, program prioriteta 3 ne može da pristupa podacima koji pripadaju prioritetnim nivoima 0, 1 i 2.

Što se tiče potprograma, ograničenje je mnogo čvršće: program na bilo kom nivou ne može da poziva programe i potprograme na drugim nivoima. Na primer, program na nivou 1 ne može zvati potprograme na nivou 0 ali ni na nivoima 2 i 3. Čak ni jezgro operativnog sistema (nivo 0) ne može, verovali ili ne, da pozove najmanje privilegovan potprogram na nivou 3!

Na ovaj način je obezbeđena prilično zaštita, ali se i dalje postavlja pitanje kako će aplikacija zatražiti uslugu operativnog sistema? Možda da mu pošalje pismo poput „moćni operativni sistemu, tvoja skromna i verna aplikacija želi da te zamoli da joj milostivo dodeliš još malo RAM-a za njen stek"? Ni takva poruka se, na žalost, ne bi mogla dostaviti jer između pojedinih nivoa postoje jaki „zidovi". Na svu sreću, na nekima od tih zidova otvorena su vrata...

Vrata (odnosno *gate*) su specijalni objekti (objekat je, kao i ranije, okarakterisan odgovarajućim deskriptorom u zoni operativnog sistema) koji obezbeđuju komunikaciju među procesima sa različitim privilegijama. Operativni sistem, na primer, može da odluči da u krugu 3 otvori jedna vrata koja pokazuju na neki potprogram nivoa 0. Obzirom da su vrata na nivou 3, aplikacija na tom nivou ih može „otvoriti", posle čega se kontrola predaje odgovarajućem potprogramu samog operativnog sistema; za vreme izvršavanja tog potprograma, proces privremeno dobija prioritet 0. Vredi primetiti da je na ovaj način omogućeno samo **izvršavanje** potprograma, a ne i njegovo čitanje, analiziranje, debagovanje... Takođe, ne može se pozvati „bilo šta" u operativnom sistemu, nego samo oni potprogrami na koje vrata pokazuju. Postoje četiri vrste vrata: *call*, *interrupt*, *trap* i *task*, čija je značenje manje-više očigledno a opisi poprilično različiti. Uvedeni su i drugi bezbednosni mehanizmi, ali verujemo da ste već na osnovu ovog opisa zaključili na kom principu sve to funkcioniše i koliko je komplikovano.

15. 80486 je obezbeđen od svega, pa i od grešaka koje se jave tokom obrade drugih grešaka.

Samo do neke granice. Naime, ukoliko se u toku obrade jednog interapta ili greške dogodi nešto jednako ozbiljno, sistem može da generiše takozvani *double fault fault*. Recimo, korisnik zahteva deljenje nulom, i čim krene obrada te greške pokaže se da na steku nema mesta za poziv odgovarajućeg programa. U tom slučaju automatski se poziva INT 08 koji treba da sredi stanje, uglavnom tako što će prekinuti izvršavanje programa. Jer, ako se tokom obrade *double fault fault*-a dogodi nova greška, mikroprocesor se blokira i čeka isključivo *reset* ili NMI – sve aplikacije koje su se prethodno izvr-

OSNOVNI TERMINI

Realni režim (*real mode*) – stanje u kome se procesor 80286/80386/80486 „budi". U takvom stanju 32-bitni procesor je „degradiran" na šesnaestobitni 8086 i izvršava sve (dobro, praktično sve) njegove programe. Može da adresira samo megabajt memorije, do ostatka se dolazi „pomoću trikova" itd.

Zaštićeni režim (*protected mode*) – stanje koje postoji počevši od procesora 80286. I dalje su „na snazi" sve instrukcije 8086, ali se sadržaj segmentnih registara prilikom pristupa memoriji interpretira na novi način. U zaštićenom režimu 80286 može da adresira 16 megabajta memorije, a obezbeđena je i zaštita zona memorije od aplikacija koje pokušavaju da pristupe onoma na šta nemaju pravo. Ovo su ujedno i osnovne pretpostavke za realizaciju višekorisničkog ili višeprogramskog operativnog sistema. Teorijski se mogu pisati aplikacije koje se, bez ponovnog prevođenja, izvršavaju i u realnom i u zaštićenom režimu, ali je takvih na tržištu jako malo (ako ih uopšte i ima).

I 80386/80486 mogu da rade u zaštićenom modu, ali dizajner operativnog sistema koji će se izvršavati na nekom od ovih mikroprocesora ima još jedan stepen slobode. Može da se opredeli za mod kompatibilnosti sa 80286 (u kom slučaju „pristaje" i na pomenuta ograničenja) ili za novi 32-bitni zaštićeni mod u kome važi novi set instrukcija i u kome su segmenti ograničeni na 4 gigabajta.

Virtuelni režim – Kada radi u 32-bitnom zaštićenom režimu, 80386/80486 može da „lansira" proces koji se prividno izvršava u realnom režimu, dakle potpuno emulira 8086. Taj proces „ne vidi" ništa izvan svog megabajta memorije, ali to ne znači da je samostalan – mogu se uvesti razni interapti koji nastupaju kada proces poželi da izađe na periferiju, upisuje u neku zonu memorije i tome slično. U istom trenutku može da se izvršava veći broj 8086 virtualnih procesa, pri čemu svaki od njih „misli" da je čitav računar njemu na raspolaganju.

Virtualna memorija – Princip koji se često koristi kod višekorisničkih operativnih sistema. Ideja je u tome da se uz korišćenjem „male" količine RAM memorije i prostora na disku simulira daleko veći RAM. Svaki program, na primer, ima utisak da radi sa 8 megabajta memorije, u sistemu je samo 4 megabajta a ima nekoliko programa. Program operiše sa fiktivnim adresama koje se, za njega potpuno transparentno, prevode u realne adrese. Kada se ustanovi da podatak koji je programu potreban nije u memoriji, nastupa interapt, izvršavanje programa se privremeno suspenduje a podatak dovodi u memoriju namesto nekog drugog koji davno nije korišćen i koji se odlaže na disk. Memorija se najčešće deli na stranice fiksne veličine, a ponekad i na segmente ili neke druge hibridne strukture. Rad sa virtualnom memorijom je uveden još na 80286 ali je pravi smisao dobio na 80386/80486. Pa ipak, za sada se retko koristi na PC računarima, ali je nezamenljiv na nekim drugim operativnim sistemima.

šavale se nelegalno zaustavljaju. Mikroprocesor, doduše, tom prilikom generiše signal *shutdown* tako da se periferije može snalaziti, pa ipak, situacija je krajnje neprijatna.

Sve ovo ne bi nikako smelo da se desi na korektno napisanom operativnom sistemu, ali u fazi razvoja treba biti veoma pažljiv. Sam 80486 će vam maksimalno pomoći tako što „realne" *double fault fault* situacije tretira potpuno automatski, pa ove greška ne nastupa – ukoliko, na primer, nastane greška, a program koji treba da je obradi trenutno nije u memoriji, najpre će biti izvršen *page fault* kako bi podaci stigli u memoriju, pa se tek onda program poziva, sve bez ikakve intervencije programera.

16. Nemaskirani interapt ima apsolutni prioritet i prekida sve što mikroprocesor radi. Nemaskireni interapt se ne može nikako onemogućiti.

Nekad bilo. Kada govorimo o mikroprocesorima 80386 i 80486, NMI ima relativno nizak prioritet. Zapravo „drugi od pozadi" – manje prioritetan od njega je jedino maskirani interapt INTR. Prve četiri prioriteta su redom (da se ne trudimo da ih prevedemo): *nondebug fault, trap instruction, debug trap for the current instruction* i *debug fault for the pending instruction*.

Ukoliko NMI i neko od ovih stanja nastupe istovremeno, prvo se obrađuje to stanje, pa tek onda nemaskirani interapt. Dok se nemaskirani interapt sasvim ne obrati (IRET instrukcija), nastupanje novog nemaskiranog interapta je maskirano, tj. onemogućeno. Kao što se vidi, konstruktori 80486 nisu imali mnogo poštovanja prema terminu *nemaskirani* interapt.

17. Intelovi mikroprocesori imaju jednobajtnu instrukciju INT 03 (kod 0CCh). Da bi neki debager napravio prekidnu tačku u programu, on jednostavno „zapamti" šta se tamo nalazilo a onda upiše 0CCh i tako obezbedi izlaz u trenutku nailaska na kritično mesto.

Još jedna od „nekad bilo" stvari. Istini za volju, instrukcija 0CCh i dalje postoji i može se koristiti na opisani način. Ipak, 80486 to radi mnogo odlje i brže – u neki od registara DR0-DR3 upišete 32 bita neke adrese, podesite još neke flegove u DR6 i DR7 i naprosto očekujete da vam se kontrola vrati čim program koji „islеdujete" stigne dotle. Program čije izvršavanje pratite se ni u jednom trenutku ne menja, a na čitavu proveru se ne troši ni malo vremena – sve teče potpuno automatski.

Uopšte, konstruktori 80386 i 80486 su uložili ogroman napor da obezbede moćne i brze alatke za pronalaženje grešaka u programima.

18. Kada radite na nekom višekorisničkom sistemu (npr. VAX), sve vaše akcije se automatski „kontrolišu" i sistem dopušta samo ono na šta imate prava. Kada radite sa PC-jem, ceo računar je vaš!

Što se VAX-a tiče, istina. Što se PC-ja tiče... Ovo je jedna od onih čudnih tvrdnji koje su i u prvoj i u zadnjoj instanci tačne, a u praksi su vrlo daleko od istine. Naime, ako upalite 80486 PC bez ikakvih rezidentnih programa (računajući tu i segmente DOS-a kao što je HIMEM.SYS, DOS=HIGH itd), zaista ste u realnom modu i možete dosta toga da radite. Ne baš sve, ali skoro sve. Takođe, u krajnjoj instanci, uz dosta znanja, truda i veštine možete da naterate računar da uradi ono što mu naredite. Ali u praksi, svaka vaša akcija je itekako kontrolisana.

Nastavlja se na strani 57

PRIČA SA ZAPADNE STRANE

Može li se istorija ponoviti? Može, može, i to po više puta, samo ako za to postoji računica. Intel nas je sa 4-bitnog doveo do 32-bitnih procesora, sprema na čak i 64-bitni procesor, a sa druge strane izbacuje čudno prikraćene procesore. Zašto i kako, a ponajviše, ima li sve to smisla – sve su to pitanja koja se postavljaju pred novog klinca u ulici – Intel i80486SX.

Dejan V. Veselinović

Kada je Intel svetu prikazao prve primerke svoga za ono vreme revolucionarnog mikroprocesora, modela i8086, krajem sedamdesetih godina, kao svojevrstan problem se postavio nedostatak jeftinih logičkih kola za 16-bitnu arhitekturu. Naravno, bilo ih je, ali su bila prilično skupa, jer se sve do tada pravilo za 8-bitne procesore. Ovaj problem se postavio kako pred samo Intel, tako i pred potencijalno najvećeg kupca tog mikroprocesora, a to je naravno bio IBM.

PROCESORSKI MEŠANAC

Intel-u nije trebalo dugo da se doseti magične formule, zadržao je unutrašnju 16-bitnu arhitekturu procesora, ali mu je sabirnicu kojom opšti sa svojom okolinom presekao na pola. Time je dobijeno prvo računarsko kopile, mešanac koji se često opisivao oznakom „16/8-bitni procesor". Proračun je bio veoma jednostavan: pošto se iznutra radi na 16 bita, procesor zadržava sve poželjne osobine, kao što su nove naredbe i moguć pristup za ono vreme neverovatno velikoj memoriji od 1 MB, a sa druge strane u svojoj okolini koristi popularna 8-bitna, lako dobavljiva i samim tim relativno jeftina TTL kola. Kako je sve to prošlo, znate i sami – otpočela je lavina.

Potpuno ista logika je nekih 8 godina kasnije dovela do pojave procesora i80386SX, verzije 32-bitnog procesora i80386 sa 16-bitnom sabirnicom. Što je najlepše, i razlozi su opet bili čisto ekonomske prirode. Ideja je (opet) bila da se upotrebom jeftinih i lako dobavljivih 16-bitnih veznih kola naprave platforme koje će sa jedne strane omogućiti lak i jeftin ulazak u svet 32-bitnog računarstva, a sa druge strane definitivno zapečatiti tržište u korist Intel-a. Ovaj drugi put, međutim, nisu uspeli ni izbliza onako kao prvi put: svet se naprosto nije oduševio ovom idejom.

PROBLEMI, PROBLEMI...

A onda se dogodilo ono nezamislivo – neko je klonirao Intel a i prošao se tim na sudu uprkos brojnim tužbama! Kao što je poznato, američka firma AMD je isklonirala procesor i80386, ali ne kao prost klon, već kao dograđen proizvod. Ne ulazeći sada u to kako i koliko, ali važno je napomenuti dve stvari: prvo, AMD je svoju verziju odmah napravio u submikronskoj tehnologiji, i drugo, umesto NMOS, odmah je upotrebio CMOS tehniku. Za korisnike, oba elementa znače procesor koji može pouzdano da radi na taktovima većim od Intel-ovih procesora i naravno nižu cenu, jer ma kako sekli, kolač se mora nekako podeliti, a pošto je Intel ipak provereni original, AMD je morao da zaigra na kartu cene kako bi sebi privukao proizvođače opreme.

Dodajući odlu i uvredu, AMD je veoma brzo isklonirao i SX procesor, opet na većim taktovima od Intel-a (Intel nudi radne taktove od 16 i 20 MHz, a AMD 16, 20 i 25 MHz). Da sve bude još gore, AMD procesor 80386 radi na 40 MHz, a uskoro se očekuje i verzija na 50 MHz; ovaj drugi je već nož u leđa Intel-a čak i po pitanju procesora 80486DX na 25 MHz – ista efektivna brzina uz manju cenu kod AMD-a. Užas! Tako nešto samo krv (dolarska) može oprati!

Posežući unazad u istoriju, Intel dolazi po treći put do magične formule – hajde da malo očerupamo 486DX i da ga nazovemo SX. Uz put ga i malo pojeftinimo, što neće biti previše teško, jer i na ovom 486DX već deremo.

Ali, kako da ga očerupamo da bude dovoljno moćan da eliminiše konkurenciju, dovoljno jeftin da bude veoma interesantan, a da ga ne obogaljimo? Drugim rečima, šta mu oduzeti a šta mu ostaviti, kako mu odrediti cenu da zaradimo a budemo konkurentni, a da pri svemu tome ne skliznemo nadole? Dilema nije bila laka, nema šta.



Slika 1 Most i486SX

Most 486SX i 486/25 - HW



	Reference	Most 486SX	Most 486/25	R/Optima 486/II
Video	1	1.85	2.16	2.47
T. disk	1	1.09	1.1	1.08
Memorija	1	1.76	2.04	2.21
Procesor	1	2.22	2.8	2.81

Procesor · Memorija · T. disk · Video

Mašinski testovi

Slika 2 Most i486SX

Most SX/20 i 386/33 - SW



	Reference	Most SX/20	Most 386/33	R/Optima 486/II
Proračuni	1	0.98	1.38	2.68
Baze p.	1	1.1	1.3	1.98
CAD	1	1.3	2.49	5.63
Grafika	1	1.04	1.85	2.08
Ob. teksta	1	1.13	1.87	2.5

Ob. teksta · Grafika · CAD · Baze p. · Proračuni

Programski testovi

Umesto klasičnog sužavanje sabirnice, Intel se opredelio za sasvim drugačiji pristup. Prvo, oborio je radni takt ovom procesoru sa početnih 25 MHz u slučaju i80486DX na 20 MHz, odnosno za -25%. Drugo, izbacio je interni modul koprocesora, čime je pojednostavio proizvodnju, samim tim je pojeftinio, a nije suštinski oštetio sam procesor.

i80486SX i dalje unutar sebe sadrži procesor 80386DX u punoj 32-bitnoj verziji, keš kontroler i82385 i 8 KB statičke keš memorije, nedostaje mu samo i80387. Manje tranzistora na silicijumu, jeftinije za proizvodnju i manje škarte – to su sleri i provereni postulati. Sa pozitivne strane, usled i dalje veoma visokog stepena integracije, procesor zadržava sve vrline većeg brata, a interna keš memorija ga čini lakim za povezivanje sa relativno sporijom memorijom (uz sada već obavezne dopunske efekte preplitanja) i brzim u radu. Bar na papiru.

...A KAKO U PRAKSI

U okviru naše već tradicionalne akcije testiranja hardvera, dobili smo jedan model računara koji koristi ovaj procesor („Most", Beograd). Ploča je baš „taze", sa oznakom proizvodnje za septembar mesec ove godine.

i80486SX izgleda u dlaku isto kao i stariji brat, sem, naravno, u smislu oznake. Ostatak ploče je dosta standardan, a opet sa nekim specifičnostima. Prva specificnost su ETEQ vezna kola (ET82C491/ i ET82C492), koja u saradnji sa jednim C&T VLSI kolom (82C206) obezbeđuju veliku vecinu vezne logike (C&T kolo kontroliše sabirnicu ka spoljnoj memoriji u utičnicama za proširenja). Pored njih, na ploči „Baby AT" formata se nalazi još svega 24 TTL kola (računajući i kontrolu keš memorije), jedno PAL kolo, 10 SRAM čipova (keš i tag RAM) i naravno 3 ROM BIOS kola.

Ploča sadrži ukupno 8 utičnica za proširenja, od kojih je jedna 32-bitna (za dopunsku memoriju) ili 8-bitna, jedna je 8-bitna, a šest su 16-bitne. Memorija se postavlja u priložen'h osam mesta za SIM module. Možete koristiti module sa 256 KB, 1 MB ili 4 MB, čime na ploču možete staviti ukupno 32 MB, a računar će, sa posebnom kartom, prepoznati i celih 64 MB. Što je najlepše, možete čak donekle i mešati SIM-ove, najmanje sa najvećim, mada je broj kombinacija ipak konačan.

Konstruktori se nisu zadovoljili samo sa internom keš memorijom procesora, pa su mu dodeli dopunskih 64 KB drugostepene keš memorije, u vidu „Toshiba TC5588P-25" kola (25 nS). Dalje, na ploči se nalaze dva umesto jednog kristala, koji se preklapaju jednim kratkospojnikom. To znači da naknadno možete ovaj procesor zameniti „pravim" i80486DX i istovremeno povećati radni takt za 25%. Nemojte misliti da je ovo ludo: treba da znate da puni 486DX na 25 MHz košta manje od numeričkog koprocesora za 80386DX na 33 MHz (o 40 MHz i više da i ne govorimo). Valjda zbog ovakvog stava na ploči nema podnožja predviđenog za koprocesor, pa čak ni za Weitek 4176. Za probu smo ugradili 8 MB RAM-a.

Ostatak konfiguracije je bio sasvim nalik standardnoj konfiguraciji preduzeća „Most": tvrdi disk je Quantum 40AT, video karta sa Tseng ET4000 video čipom i 1 MB memorije, i sve što uz to pripada.

MERENJA

Ovo nam je, naravno, bio najinteresantniji deo posla. Pored interesovanja za stvarne efekte ovog hibrida, bilo nam je veoma pogodno i to što smo na raspolaganju imali rezultate merenja raznih drugih mašina.

U odnosu na prosek rezultata, mašina na bazi procesora i80386 na 33 MHz sa spoljnom keš memorijom, pokazalo se da i486SX uopšte ne zaostaje. Štaviše, nešto malo je brži od njih, mada je ta razlika zaista samo akademska (1-2%).

U mašinskim testovima je pokazao nešto gore rezultate, ali ovo nije sasvim realna slika, jer se dosta veliki broj tih testova oslanja na radni takt. Pošto su programski testovi daleko objektivniji (jer uzimaju istovremeno u obzir sve), njihove rezultate treba smatrati zaista merodavnim, a po njima, ovaj procesor je oko 4% brži od i386 na 33 MHz.

Kada se uzme u obzir razlike u količini keš memorije, dolazimo do zaključka da su jače konfiguracije brže taman za onoliko koliko im je i radni takt veći, plus one 2-5% na veću keš memoriju. Time se potvrđuje teza da u funkcionalnom smislu za rad bez numerike koja zahteva koprocesor, i486SX zaista ni po čemu ne zaostaje za većim bratom.

Korisna adresa

"Most" računari
Kralja Bodina 2, YU-11000 Beograd
Tel. (011) 426-144, Fax. (011) 426 144

KUPITI ILI NE

Odgovor je veoma jednostavan: sve zavisi od cene i vrste posla kojom se bavite. Ako je cena ovakve ploče ista kao i druge koja koristi i386 na 33 MHz, ovu drugu jednostavno preskočite i uzmite onu sa i486.

Drugi aspekt je, istina, vezan za baš ovaj primerak ploče, ali čak i tako zaslužuje malo pažnje. Mogućnost naknadne ugradnje „pravog" i486DX uz povećanje radnog takta za 25% vam otvara vrata daljem širenju i bavljenju već i pojačom numerikom i CAD-om, uz prednost da je vaša investicija razvučena na neko vreme.

Naravno, i486DX na 33 MHz je dobrih 80% brži, pa ako vam je brzina izuzetno važna, preskočite sve i pravac na njega. Ali, ako se bavite DTP-om, jako se zamislite pre nego što kupite 386 na 33 MHz: ovo je bolji posao, pre svega zbog dvostepene keš memorije, od koje potiče razlika u brzini od 5-9% u korist ove ploče. Sa nekom boljom video kartom, ova razlika bi se mogla povećati u odnosu na prosek.

POMUTITI RAČUNE

Ovaj procesor je ubačen u igru samo zato da bi AMD-u i drugim kloneraima pomutio račune. Nije uopšte isključeno da sam Intel odusta ne od daljeg razvoja i386 u korist i486 i naravno i586, zašto da ne? Na ceni će uvek gubiti u odnosu na konkurenciju, a tehnologiju četiri osam šestice za sada imaju samo oni, iako ima indicija da bi se i to veoma brzo moglo promeniti.

Na 20 MHz, ovaj čip se srčano bori sa klonovima na 40 MHz, a verzija na 25 MHz bi se borile sa klonovima na 50 MHz. U svemu tome, Intel ima prednost baš zbog manjeg radnog takta: što je on manji, lakše je sinhronizovati delove sa procesorom i naravno, lakše se radi sa jeftinijom memorijom, problem sa kojim se takt od 50 MHz i te kako treba boriti.

Dakle, ako pogode cenu – svaka čast, brzina i za malo manje bogate. Isto važi i za „Most".

OSMORICA VELIČANSTVENIH

Zbog specifičnih prilika na domaćem tržištu, na kome ponuda nikada nije bila jasno definisana, retko smo se upuštali u velike uporedne testove hardvera. Ovoga puta uporednim megatestom obuhvatili smo ponudu dve poznate beogradske firme, Game i Mosta, a u sledećim brojevima će biti neko drugi – poziv je otvoren za sve.

Dejan V. Veselinović

Pre nekog vremena uputili smo poziv raznim ponuđacima opreme da nam na testiranje ustupe neke svoje proizvode. Ali ne po jedan primerak, kako je postalo uobičajeno, već po nekoliko komada, smatrajući da je reprezentativnije i za njih i za nas. Tako potencijalni kupci stiču bolji pregled cele ponude, a ne samo ovog ili onog modela. Sem toga, ako neko nudi jedan model koji je izuzetno uspešan, to nije garancija da su i ostali ili drugi modeli jednako uspeli u smislu integracije.

MOST, Beograd

Preduzeće „Most" iz Beograda je bilo među prvima koji su se odazvali na naš poziv, naravno, ne računajući probni uzorak prodavaca. Na testiranje su podneli nekoliko modela, koji praktično čine okosnicu njihove ponude, u tom smislu što se radi o baznim modelima na osnovu kojih se zamenom delova dobijaju dopunske varijacije na istu temu. Ovi modeli su oni za koje ćete dobiti osnovne cene kada se budete raspitivali za njih.

"Most" inače nudi čitav niz dodataka odnosno povećanja u odnosu na osnovne konfiguracije koje smo ovde testirali, od većih diskova do drugih stvari, kao što su strimeri, mreže, programska rešenja i tome slično.

Jedna zajednička osobina svih matičnih ploča koje „Most" nudi je da potiču od istog proizvođača, tajvanske firme „Soyo". Oni su opet počeli da ugrađuju novu generaciju SIM postolja koja umesto plastičnih imaju metalne zadrške za module. Sitnica, ali nama se jako dopala, isuviše dobro znamo koliko su plastične zadrške (kukice) podložna lomljenju.

MOST SX

Procesor, takt: 80386SX, 20 MHz
Memorija: 2 MB
Tvrdi disk: Quantum, 42 MB
Video podsistem: VGA/EVGA/SVGA
Ocena: **(M)** 63,2% **(El)** 38,1% **(P)** 10

Model smo dobili zapakovan u veoma ukusnom „mini tower" kućište, koje spada među do sada najbolja koja smo videli. Veoma je čvrsto, pa čak i lepo na oko. Deluje ubedljivo i ni malo se ne giba pri normalnom prenošenju.

Tastatura je standardan „Cherry" model u našoj verziji sa YUSCII znacima. Potpuno je tiha, ali nije isuviše mekana i u radu ostavlja pozitivan osećaj. Iako nije baš idealna po jednom ocenjivaču, drugi je imao bolje mišljenje o njoj.

Monitor nosi naziv „Tystar", a radi se o kolor monitoru sa prečnikom ekrana od 14 inča, sposobnim da radi do rezolucije od 800×600 bez i 1024×768 sa preplitanjem. Sasvim je pristojan, mada ni po čemu izuzetan, oba ocenjivača su ga smatrala nekakvim prosekom za današnje pojmove, pa su ga tako i ocenila. Pomalo izobličuje sliku pri jakim bojama (po uglovima), mada istini za volju, ovo je redovna pojava u ovoj klasi cene. Izmereni nivo elektrostatičkih zračenja je bio zadovoljavajuće mali, mada za ove pare ima i boljih.

Tvrdi disk je poreklom od poznate i veoma cenjene firme Quantum, mada se radi o sada već pomalo zastarelom modelu „ProDrive 40AT". To ga doduše nije ni malo sprečilo da radi čak veoma dobro, sa solidnim rezultatima kojih nema zašto da se stidi. Disk je zadovoljio sve kriterijume brzine, a imajući u vidu renome i kvalitet proizvođača, koji u svetu vodi po izuzetno malom broju vraćenih diskova (ispod 0,5%), treba očekivati godine pouzdanog i brzog rada od njega.

Matična ploča ima format „baby AT" ploče i napravljena je uz obilato korišćenje tehnologije površinskog lemljenja (SMT). Svi delovi nose propisne oznake za radni takt, čime se obezbeđuje pouzdan rad tokom celog životnog veka. Važna kola su od firme OPTI, sve prisutnije na svetskom tržištu, a BIOS je poreklom od jednako poznate i cenjene firme AMI. Proizvođač ploče je tajvanska firma SOYO, čije reklame možete naći po svetskim časopisima. Na samu ploču možete staviti do 8 MB memorije, mada BIOS prepoznaje svih 16 MB. Najzad, ploča je opremljena i statičkom keš memorijom od 32 KB, čime se procesoru omogućava da 90+% radi bez stanja čekanja.

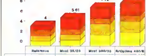
U odnosu na apsolutnu referencu (vidi prilog), ovaj računar je pokazao ukupni mašinski indeks performansi od 1,40, pre svega zbog izuzetno brzih video performansi. One su opet rezultat VGA kartice sa Tseng ET4000 video čipom, ako uzimate ovaj računar preporučujemo vam baš ovu kartu, dodiće te nadprosečne rezultate.

U programskim testovima, računar se pokazao jedva 10% (indeks 1,10) bržim od naše reference, iako radi na 25% većem taktu. To je sasvim uobičajeno u ovakvim slučajevima, jer se radi o 16-bitnim naredbama koje stariji procesor 80286 izvršava brže od svoje 32-bitne braće, naravno, on zato po definiciji ne može koliko veća braća.

U celini gledano, radi se o jednoj veoma pristojnoj mašini koja je u odnosu na sam ulazni nivo 32-bitne klase pomerena za kvalitet video podsistema (VGA umesto Hercules-a). Nama se došla dopala, pa iako nije šampion

Slika 1

Most SX/20 i 386/33



	Referenca	Most SX/20	Most 386/33	R/Optima 486/II
Video	1	2.26	2.23	2.47
T. disk	1	1.07	1.06	1.08
Memorija	1	1.08	2.21	2.21
Procesor	1	1.2	2.23	2.81

Procesor, Memorija, T. disk, Video

Slika 2

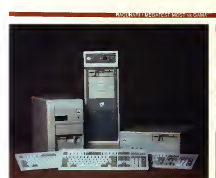
Most SX/20 i 386/33 – SW



	Referenca	Most SX/20	Most 386/33	R/Optima 486/II
Proracuni	1	0.96	1.36	2.66
Baze p.	1	1,1	1.3	1.96
CAD	1	1.3	2.49	5.63
Grafika	1	1.04	1.66	2.08
Ob. tekste	1	1.13	1.37	2.5

Ob. tekste, Grafika, CAD, Baze p., Proracuni

Hardverski i softverski testovi mašina firme „Most"



Lepeza računara „Most"

brzine, predstavlja dobar osnov za niz konsni ka umerenijih (mada ne i baš skromnih) zahte va. Idealan partner – mini „tower" kucište kao stvoreni su jedno za drugo. Dobar posao.

MOST 386-33

Procesor takt 80386DX 33 MHz
Memorija 4 MB
Tvrdi disk Quantum, 42 MB
Video podsistem VGA/EVGA/SVGA
Ocena **(M)** 63,2% **(El)** 48 7% **(P)** 64%

Pošto je jedina razlika između ovog i pret hodnog modela različita matična ploča posve tićemo pažnju praktično samo njoj.

Kao i u prethodnom slučaju radi se o pro zvodu firme „Soyo", opet sa OPTI čipovima opet sa AMI BIOS-om i opet sa keš memorijom u vidu statičkih čipova ukupnog kapaciteta 64 KB. Međutim za razliku od modela SX ovaj model je bio opremljen sa 4 MB sistemske ma morije što kao ideju pozdravljamo suludo je kupovati ovakav računar sa svega 1 MB i terati ga da na silu boga glumi veoma brzi XT. Sa 4 MB on je već sasvim lepo u stanju da iskonsti niz prednosti čak i od takvih programa kao što je MS Windows gde se razlika u memoriji i te kako oseća.

Na mašinskim testovima ovaj računar je dao indeks od 1 93 ili gotovo duplo brže od reference. Naravno razlog ovome je tvrdi disk, čiji indeks iznosi svega 1 05 što znači da su indeksi svih drugih testova bili najmanje 2 21 ili veće u odnosu na referencu.

Na programskim testovima ovaj računar je podbacio u odnosu na konkurenciju postigavši indeks od prilično skromnih 1 84 definitivno is pod očekivane razlike. Posebno se namučio na testovima obrade teksta, gde je izvukao in deks od svega 1 37. Pošto je drugo najveće odstupanje od konkurencije bilo na testovima unakrsnih proračuna dolazimo do zaključka da uprkos keš memoriji ova ploča se u praksi muči sa radom u memoriji da paradoks bude ve ći na mašinskim testovima memorije je posti gla izrazito natprosečan indeks od 2 21. Eto još jednog primera kako ne treba verovati ni jednom pojedinačnom testu.

U celini ovaj računar dosta zaostaje za konkurencijom odnosno prosekom svoje kla se. Nije on loš samo nije posebno brz. Istini za volju ovo je jedina ovakva mašina koju smo testirali sa novim DOS-om 5 0 sve ostale su imale DOS 3 3 ili 3 31 za sada još uvek ne možemo da tvrdimo ali čini nam se da bi i tu, uprkos logici mogao biti razlog ovom (blagom) zaostajanju.

Inače graškom u isporuci dobavljača „Most" nije na vreme za ovu rundu dobio ploče sa procesorom AMD na 40 MHz savetujemo vam da pričekate da i njih probamo razlika u ceni ne bi trebalo da bude velika a razlika u brzini bi trebalo da opravda razliku u ceni. Razmislite ili o ovoj drugoj ploči ili o modelu 486SX.

MOST 486SX-20

Procesor takt Intel 80486SX 20 MHz
Memorija 4 MB
Tvrdi disk Quantum 42 MB
Video podsistem VGA/EVGA/SVGA
Ocena **(M)** 63 2% **(El)** 39 1% **(P)** 72%

O ovom procesoru smo pisali malo opširni ja drugde u ovom broju „Računara". Žalo ne bismo ponavljali sve što je već rečeno a isto važi i za ostale delove sistema koji su isti kao i za model SX.

U mašinskim testovima ovaj računar je dao indeks od 1 69 u odnosu na referencu zaostajući za procesorom 80386DX na 33 MHz. Ovo ukazuje na ugrađene i na žalost neizbežne mane mašinskih testova video, a to je oslanjanje na sam radni takt. Recimo u procesorskim testovima 386 na 33 MHz je dao indeks od 2 23, a 486SX indeks od 2 22 – samo 1% sporije. Sa istim diskom 486SX je dao indeks od 1,09 a 386DX na 33 MHz indeks od 1,05.

U programskim testovima samo na osnovu različite matične ploče 486SX je dao indeks od 1 72 (prema 1 64 za 386DX na 33 MHz). Kao i u gornjem slučaju računar je „zaglavio" na testovima obrade teksta gde je stao na tačno polovini vrednosti indeksa naše relativne reference (486DX na 25 MHz) uprkos drugostepenoj keš memoriji. Za uzvrat pokazao se osetno bržim od 386 na 33 MHz u testovima unakrsnih proračuna sa indeksom od 1.71 prema 1 38.

NOVO U OCENJIVANJU

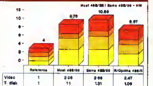
Primetićete nove oznake na početku svakog opisa mašina. Od sada na dalje počećemo da ocenjujemo konkretnim brojkama dosta toga što smo do sada samo opisivali. Da biste se snašli evo malog objašnjenja: **(M)** označava mehaniku, a to su kućište monitor i tastatura; ocenjuje se ukupno 9 elemenata sa po najmanje 5 kategorija, a rezultat se daje kao procenat. **(El)** označava elektroniku, a to su matična ploča i sve periferije unutar kucišta, rezultat je dat kao procenat od idealnog. **(P)** označava performanse, izražene kao prost procenat iznad nivoa apsolutne reference. Cene usled njihovog prelaska na dnevni režim smo preračunali u nemačke marke kako bi im obezbedili kakvu-takvu aktuelnost dok broj ne izađe iz štampe. Najzad (Ig) označava indeks garancije, pri čemu je indeks od 1 jednako periodu od 12 meseci.

Neku relativnu „vrednost" mi nismo izračunavali jer se svaka vrednost sastoji od objektivnog i subjektivnog dela, prvo možemo koliko-toliko objektivno oceniti a drugi je stvar svakog čoveka ponaosob. Da li vam je sve jednako važno ili neki aspekt ima primat nad drugima to mi ne znamo i ne možemo znati. Primera radi subjektivna vrednost se može odnositi na lični stav o značaju nekog elementa ukupne ponude ili pak na tzv percipiranu vrednost, po kojoj proizvod dobro poznate firme po definiciji vredi više od manje poznate firme a samim tim nužno i košta više (što može ali i ne mora biti tačno). Tu odluku dakle prepuštamo vama a dajemo vam „duševne hrane" da je sami donesete.

Pošto neki to neće želeti dajemo i neku našu ocenu vrednosti računara. Napominjemo da ona služi isključivo kao odraz našeg viđenja mašina i prodavca i da nije obavezujuća. Preporuke ćemo davati u dva oblika. Prvi stepen je ocena „Nadprosečno" koja označava proizvod koji se po nekoj osobini odvaja od mase sličnih ali ipak nije na nivou izuzetnog proizvoda. Drugi i najviši stepen je ocena „R/izbor" za „izbor „Računara" odnosno model ili proizvod koji preporučujemo.

I na kraju pitanje reference. Usled stalnih pomeranja cena uglavnom na dole u svetu i uglavnom na gore kod nas odlučili smo da sva merenja obavljamo u odnosu na standardnu referencu. Ona se sastoji od jedne ploče sa „Headland" G2 čipom i procesorom 80C286 na 16 MHz bez ikakvih stanja čekanja jednog Quantum LPS52A tvrdog diska jedne „Morse KP 1024 TVGA" video karte i Philips crno-belog VGA monitora. To je apsolutna referenca postoji i relativna referenca koja je opisana u „Računarima" Br 75 st. 8-9. Prva mašina se manje-više ista takva može nabaviti od „Game Electronics Beograd" a druga od „Mlacom"/"Mlakar & Co". To su za sada najbolje mašine u svojim klasama a koliko će izdržati na tim mestima živi bili pa videli.

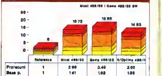
Slika 3



	Referenca	Most 486/33	Gama 486/33	R/Optima 486/II
Video	1	2.26	2.69	2.47
T. disk	1	1.1	1.31	1.08
Memorija	1	2.69	3.19	2.21
Procesor	1	3.75	3.70	2.81

Uporedni hardverski...

Slika 4



	Referenca	Most 486/33	Gama 486/33	R/Optima 486/II
Proračuni	1	2.66	3.49	2.66
Baze p.	1	1.41	1.62	1.98
CAD	1	7.36	8.01	6.63
Grafika	1	3.3	2.79	2.05
Ob. teksta	1	2.33	2.69	2.5

...i softverski test najjačih mašina u familiji "Most" i "Gama"



Matična ploča "Gama" 386DX 40

Sve u svemu, ovo je dosta brza mašina, brža no što bi se očekivalo od relativno sporog rednog teksta. Kao takva, predstavlja realnu alternativu „navijenim" procesorima 80386 od Intel-a i AMD-a, bar u ponudi „Mosta". Naravno, ukoliko cena bude razumno veća od starije tehnologije.

Napominjemo da ove ploče ima i jednu posebnost, ukoliko poželite da naknadno ugradite numeričke sposobnosti, izvadićete procesor i zameniti ga „normalnom" procesorom 80486DX, nakon čega ćete prebaciti jedan kratkospojnik i istovremeno ubrzati računar na 25 MHz (ili za 25%), čime ćete dobiti „pravi" 486. Ovaj podatak takođe treba da imate na umu prilikom odlučivanja o kupovini: smatramo da u tom kontekstu ovakva ploča predstavlja bolje rešenje od ploča sa procesorom 80386DX.

Razmislite o ovoj mašini, dobra je.

MOST 486DX-25

Procesor, takt: Intel 80486DX, 25 MHz
Memorija: 4 MB
Tvrdi disk: Quantum, 42 MB
Video podsistem: VGA/EVGA/SVGA
Ocena: **(M)** 65,6%, **(EI)** 79,9%, **(P)** 171%

Sa istim komponentama kao i u prvom slučaju, ali sa zamenom ploče, eto novog modela. Ali, ovaj je već nešto sasvim drugo. Kao prvo, ima celih 256 KB keš memorije sa jedne, a sa druge strane, može da primi memorijske module od 256 KB, 1 MB i 4 MB, ako biste upotrebili ove poslednje, na ploču biste stavili celih 64 MB memorije. Time bi ovaj računar mogao da postane čak veoma ozbiljan server ili UNIX mašina. Sa druge strane, možete mešati module čak i one sa 256 KB sa onima od 4 MB, uz uslov da manji moraju doći prvi i naravno sa nekim drugim ograničenjima u broju kombinacija.

U mašinskim testovima, ovaj računar je postigao indeks od veoma zdravih 2,03 u odnosu na referencu, što je ipak nešto ispod nivoa naše relativne reference (sa indeksom od 2,14). Interesantno je da je referenca bila brža za po 1-2% u svemu, ali je pokazala osetno bolje rezultate u video podsistemu (2,47 prema 2,16 za „Most").

I u programskim testovima, naša relativna referenca je malo povukla (2,97 prema 2,71 za „Most"). Većina te razlike potiče sa podsistema diska, gde je „Most" postigao indeks od 1,36, a naša relativna referenca 1,96. Naravno, sa bržim diskom, koja „Most" nudi kao deo svoje redovne ponude, ova razlika bi se smanjila, a možda i preokrenula.

Ovo je jedna zaista dobra mašina, koja deluje krajnje ubedljivo. Pošto je malo verovatno da biste je kupili sa ovako malim diskom, a pošto veći diskovi po definiciji imaju i bolje performanse, smatramo da možete računati na mašinu negde na nivou naše relativne reference.



Matična ploča "Gama" 486DX 33

Napominjemo da ova i naredna ploča umesto uobičajenih 8 utičnica raspolažu sa po 12 (!) utičnica, od kojih su sva 16-bitna. Podeljene su u dve grupe – osam ih je na uobičajenim mestima, a dopunske 4 utičnice su pomerene van standarda. Njih možete i sada koristiti za sve kartice koje nemaju potrebe da gledaju van kućišta, a da bi i one bile pristupačne spolja, potrebno je imati posebno kućište napravljeno za upravo ovakve ploče. Ovaj pristup proširenjima jasno ukazuje na orijentaciju ploče ka profesionalnom segmentu pre svega servera, koji se obično nakrcaju raznoraznim dodacima pa uvek nekako nedostaje još jedna ili dva utičnice.



Matična ploča "Gama" 386SX 20

Svakako uzmite u obzir ovaj model.

MOST 486DX-33

Procesor, takt: Intel 80486DX, 33 MHz
Memorija: 4 MB
Tvrdi disk: Quantum, 42 MB
Video podsistem: VGA/EVGA/SVGA
Ocena: **(M)** 65,6%, **(EI)** 79,9%, **(P)** 236%

Najzad, i „Most-to-the-Max" model. Matična ploča je praktično ista kao i na manjem modelu, sem što su delovi, odnosno procesor i čipovi, uredno deklarisani za veći radni takt.

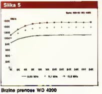
U mašinskim testovima, ovaj model je postigao opšti indeks od 2,44 prema apsolutnoj referenci, odnosno 1,14 prema relativnoj referenci. Naravno, u čisto procesorskim testovima, ovaj računar je zabeležio indeks od 3,75, odnosno gotovo četiri puta brža. Verujte, ovo nisu prazne brojke, ovo su razlike koje se i te kako osećaju.

U programskim testovima, računar je postigao indeks od 3,36 prema apsolutnoj i 1,13 prema relativnoj referenci. Opet, glavni uzrok ovako maloj razlici u performansama je disk, koji ma kako dobar bio, opet nije više u onoj klasi računara sa kojima se poredi (mada, to važi i za cenu).

Nema spora oko toga da je ovo jedna veoma, veoma brza mašina, koja po svakoj logici prosto vapi za bržim diskom. Ne usuđujemo se ni da pomislimo kako bi ovo radilo sa nekim inteligentnim keš kontrolerom tvrdog diska.

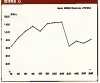
Ako vam treba velika računarska snaga, ovo je mašina za vas.

Slika 5



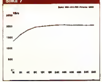
Brzine prenose WD 4200

Slika 6



Brzine prenose Quentum LPS105a (386-20SX)

Slika 7



Brzine prenosa Piranha 4200

Adrese „Most" Kralje Bodine 2, 11000 Beograd tel (011) tel (011) 426-144 fax (011) 426-144

GAMA, Beograd

"Geme" iz Beogrede nije novi gost ne stranicame „Računare", keo što nije nikakav početnik u opštim okvirima Radi se o jednoj od firmi koje, ber u Beogredu, uživaju izuzetan renome po pitanju performansi i pouzdanosti e otud nužno i cene Nije nikakve tajne de je „Game" jeden od doste skupih ponuđeče, ali se uvek znelo i de je njihove ponude veoma jaka, uvek među najjačima i znalo se da su njihove mašine „performence" mašine, uvek među najbržima

Iz objektivnih razloge ze ovaj put smo pogledeli „samo" donji deo ponude e ze gornji deo ćemo morati malo de se strpimo zahvaljujući već legendarnoj likvidnosti naših baneka

GAMA 386SX-20

Procesor takt Intel 80386SX, 20 MHz
Memorija 2 MB
Tvrdi disk Quentum, 105 MB
Video podsistem VGA/EVGA/SVGA
Ocene (M) 66 6%, (Ei) 44 9% (P) 19%

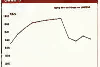
Ovaj model predstavlja „Geminu" ulaznicu u svet 32-bitnih računare Radi se o relativno klasičnoj mešini, ali, keo što ćemo videti, na višem nivou performansi od većine konkurencije

Matična ploča je relativno tipična ze ovu klesu ber spolja posmatrano, tu su svi čipovi namenjeni radu na deklarisanom taktu 32 KB keš memorije osam podnožije za SIM module i mesto za jednu dodatnu kartu namenjenu memorijskom proširenju (koja se može koristiti i keo obična 8-bitna uličnica)

Tvrdi disk je već dobro poznat model Quentum LPS105A, tenek disk se veoma malom potrošnjom struje i se zelete odličnim performanseme Kontroler diskova i diskete se nalazi na istoj karti kao i sve veze (jedne paralelne i dve serijske) Iako ovo rešenje ne volimo mnogo, izgleda de se ovde radi o najnovijoj generaciji ovakvih kartica Izgleda krajnje ubedljivo, e umesto budi-bog-se-name čipove koristi između ostalih i „Western Digital" VLSI kola, radile je besprekorno

Video podsistem se sastojao od jednog Philips monohrometskog VGA monitora i jedne nezovi Philips-ove VGA kartice Monitor ima oznaku 4BM2790 e radi se o novijoj verziji starog znenca koje u odnosu na prethodnike

Slika 5



Brzine prenosa Quentum LPS105a (386-33C)

ima samo jednu bitnu razliku e to je preklopnik ze normalan ili obrnut prikez (bela slova na crnoj pozadini ili crna slova na beloj pozadini) Zadržane su sve pozitivne osobine ovog monitora, e posebno veoma stabilne slike puno iskorišćenje površine ekrana i sposobnost prikazivanje velikog broja vidljivih nijansi sivog Od mane, uvedene je samo jedna – po name, bezobrazno je skup (oko DEM 430 u Nemačkoj, u poređenju sa sličnim tajvanskim proizvodima koji košteju oko DEM 230-250)

Video karta je zapravo generički proizvod firme „Paradise", dela imperije firme „Western Digital" mora da su je neverovatno jeftino prodali Philips-u, jer koristi sade već prevaziđen i zastareo PVGA-1 video čip Na karti se nalazi 256 KB video memorije, što me malo ze rad nevisokim rezolucijema, ali sasvim zadovoljava u radu se ovim monitorom

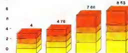
Najzad kućište je standardno „Gamino" „Baby AT" kućište koje se oduvek odlikovalo strukturnom krutosti (zbog dve poprečne šipke iznutra) i sasvim pristojnog čak simpatičnog izgleda Tastature je standardna „Cherry" YUSCII ili na zahtev ASCII pristojna ali ni po čemu izuzetna

U mašinskim testovima, ovaj model je postigao indeks od 1,19 što je dobar rezultat za ovakve mašine ali je ispod nivoa ponude „Mosta" (1,40) Do podbacivanja je došlo zbog pomenute VGA karte, koja je ekonomski kompro-



Lepeza računara "Gama"

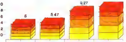
Slika 9



	Referenca	Gama SX/20	Gama 386/33	Gama 386/40
Video	1	1 26	2 42	2.85
T. disk	1	1.06	1.08	1 36
Memorija	1	1 2	1.89	1 7
Procesor	1	1 22	2.28	2 62

Procesor · Memorija · T. disk · Video

Slika 10



	Referenca	Gama SX20	Gama 386/33	Gama 386/40
Proracuni	1	0.94	1 56	1.64
Baze p	1	1.11	1 33	1 47
CAD	1	1 26	2 57	2 84
Grafika	1	1	1 84	2.24
Ob. teksta	1	1.14	1 85	2.2

Ob. teksta · Grafika · CAD · Baze p · Proracuni

Hardverski... i softverski testovi mašina firme "Gama"

miš – ako uzimate VGA, doplatite za „Morse KP1024 TVGA" karticu koja jeste najskuplja, ali je i najbrža.

U programskim testovima, opet zbog sporosti video karte, ovaj model je zaradio indeks od 1 09 („Most" 1 10). Za uzvrat, pokazao je veoma dobre rezultate na radu sa diskom, što je i bilo za očekivati.

Radi se o dobroj mašini, ali mašini koja je hendikepirana sa dva faktora – sporom video kartom i malo memorije. Pošto je bila opremljena sa 2 MB u vidu dva SIM modula, računar nije mogao da preplиće, već je bio ograničen na stranični pristup (page mode). Ako ste kupac, uzmite ga ili sa 4 MB (dobićete oko 5-6% na brzini rada) ili sa 2 MB u vidu SIM modula od po 256 KB, čime ćete dobiti čak četvorostruko preplitanje, a obavezno se opredelite za brzu (i skuplju) video kartu. Sa tim izmenama indeks će mu se veoma osetno popraviti za 15-20%, čime dobijate već natprosečnu mašinu.

Ovakav kakav je, dobar je, ali ni po čemu nije izuzetan.

GAMA 386DX-33

Procesor, takt: AMD 80386DX, 33 MHz
Memorija: 4 MB
Tvrdi disk: Quantum, 105 MB
Video podsistem: VGA/EVGA/SVGA
Ocena **(M)** 66 6% **(El)** 53 4% **(P)** 64%

U odnosu na prethodni model, ovaj računar ima, naravno, drugačiju matičnu ploču: umesto procesora Intel 80386SX, ova ploča se oslanja na procesor AMD 80386DX na radnom taktu od 33 MHz. Tu je i keš memorija od 64 KB i mesto za 8 MB u SIM modulima, moguća je kombinacija modula od 256 KB i 1 MB. Veznu logiku obezbeđuju OPTI VLSI kola, a ROM BIOS je od firme AMI.

Takođe za razliku od prethodnog modela, ovaj računar je bio opremljen pomenutom kojа je potpuno nalik na onu koju koristi i „Most" (Tseng ET4000 video čip, 1 MB video RAM-a), što se naravno pozitivno odrazilo na rezultate video testova. Svi ostali delovi su bili sasvim isti, uključujući i tvrdi disk (Quantum LPS105A).

Nova kartica vapi i za novim monitorom, u ovom slučaju radilo se o (opet) Philips SVGA modelu „Brilliance". O njemu smo već ranije pisali pa ćemo ga samo ukratko opisati kao veoma dobar model u svojoj klasi, pa čak i u malo višoj od svoje klasi. Odlikuje se veoma dobrim bojama i veoma niskim stepenom elektro magnetskih zračenja. Slika spada u vrh svoje klase, sa veoma malo konkurenata, što se posebno odnosi na mala ugaona izobličenja i dosta nizak stepen razlivanja boja. Zato je i skuplji od proseka svoje klase, što mu mi ne zameramo, jer odavno tvrdimo da na monitorima ne treba štedeti – imate samo jedan par očiju.

Sasvim očekivano, računar je postigao odličan indeks u mašinskim testovima od 1.99; ovo se mora smatrati odličnim u odnosu na apsolutnu referencu jer obuhvata i operacije sa diskom, koje nikada ne mogu biti proporcionalne razlici u brzini samog procesora. Ovo je, sa druge strane, nekih 3% brže od sličnog modela firme „Most".

U programskim testovima „Gama 386 33C" je postigao već veoma dobar indeks od 1.85, što je već 12 8% brže od modela „Most" (indeks 1.64). Inače, ovo je i dosta dobra korelacija između mašinskih i programskih testova, ne baš sasvim standardna pojava. Potiče od dobre integracije sistema, elemenata koji nikada ne treba olako odbaciti, mada je ideal da ako razlika postoji, u celini bude u korist programskih testova.

Odličan model – slobodno ga kupite.

GAMA 386DX-40

Procesor, takt: AMD 80386DX, 40 MHz
Memorija: 4 MB
Tvrdi disk: Western Digital, 210 MB
Video podsistem: VGA/EVGA/SVGA
Ocena **(M)** 68 6% **(El)** 60,3% **(P)** 114%

Ovaj model je potpuno isti kao i prethodni, sem u dva elementa: matična ploča je opremljena odgovarajućim delovima za povećani radni takt, ovaj put na 40 MHz, opet sa AMD procesorom.

Druga razlika se odnosi na tvrdi disk. Ovaj put se radi o novom modelu čuvene firme „Western Digital", sa zvučnim imenom „Piranha 4200". Nije nas ugrizao, ali nas jeste impresionirao nekim svojim osobinama. Radi se o disku sa formatiranim kapacitetom od 210 MB, koji čak i na prvi pogled deluje veoma ubedljivo, sasvim u duhu proizvoda ove firme. Pored dobre izvedbe, ovo je najbrži disk sa kojim smo se do sada susreli, kako po brzini slučajnog pristupa koja je ispod 14 mS, tako i po brzini prenosa, koja je neverovatna (vidi grafikon). Ne samo što je izuzetno ujednačena, nego se nalazi i na neverovatnim nivoima, u proseku na gotovo 2 MB/s (!!!).

Probali smo da ga zbunimo, ali bez uspeha, ma kako ga podesili, drži taj nivo prenosa pa drži. Pride spada u novu generaciju diskova sa malom potrošnjom (oko 8 VA), tih je i začudo jeftin (u kontekstu cene po megabajtu) do te mere da glatko preuzima prvo mesto na top listi sa veoma ubedljivim vođstvom (od modela Fujitsu 180 MB). Dakle, redak spoj osobina – velika brzina, ogromna brzina prenosa, veliki kapacitet i mala cena, ostaje da se u praksi vidi koliko je pouzdan.

U mašinskim testovima, ovaj model je postigao indeks od 2.13, u poređenju sa recimo indeksom od 2 14 za našu relativnu referencu, koja koristi procesor 80486 na 25 MHz. Istina, ovakva bliskost je postignuta pre svega odličnim tvrdim diskom (indeks 1 36 prema 1 08), ali – šta je, tu je, tako jeste.

Na programskim testovima, ovaj model je pokazao svoje pravo lice, postigavši željeno viši nivo nego na mašinskim testovima (2 14 prema 2,13). Naravno, programski testovi su doveli do pravog stanja stvari, jer su pokazali prave odnose između procesora. Recimo, na testu obrade teksta, ovaj model je zaradio indeks od 2 20, a naša relativna referenca indeks od 2,50, a i u ostalim testovima su dobijeni slični rezultati, pa proizilazi da je relativna referenca ipak brža za nekih 36 7%.

Sve u svemu, ovo je računar koji ima niz vrlina, od kojih je najveća, naravno, cena. U kontekstu performansi, brži je od 486SX mašine za nekih 24 4%, a jeftiniji je od nje, sporiji je od 486 na 25 MHz, ali je osetno jeftiniji od te ploče. Sve ovo važi pod uslovom da vam ne treba numerički koprocesor; ako vam on treba, kupovina nekog modela sa procesorom 80486 na 25 MHz će se pokazati kao em brže, em jeftinije rešenje.

Sa druge strane, ako se bavite, recimo, stonim izdavaštvom, a budžet vam nije baš u najboljem stanju, ovo je idealno rešenje. Isti važi i za manje servere u manjim lokalnim mrežama. Sa više memorije, radio bi koji procenat brže zbog preplitanja, a i keš memorija je proširiva do 256 KB – idealno za memorijski intenzivna aplikacije.

Od „Game" smo dobili informaciju da će najverovatnije uskoro odbaciti model na 33 MHz u korist modela na 40 MHz, pre svega usled male razlike u ceni, a solidne razlike u brzini. Čak i da do toga ne dođe, dobro razmi-

Nastavlja se na strani 63

I PERO I RAPIDOGRAF

Štampač, laserski ili matrični, neophodna je alatka svima koji se bave obradom teksta na bilo koji način. Korisnici kojima su, pre svega, potrebni crteži i slike, moraju ili da pristanu na sva ograničenja štampača ili da nabave ploter. Ipak, postoji i treća mogućnost – štampač koji i štampa i crta dovoljno dobro da zadovolji obe profesionalne potrebe.

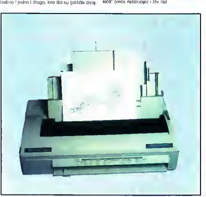
Marko Kirić

Iako većina savremenih 24-pinaca postiže rezoluciju od 300 ili čak i 360 tačaka po inču, njihov otisak se već na prvi pogled razlikuje od otiska napravljenog na laserskom štampaču – na njihovu štetu. Laserski štampači, sa druge strane, ostaju bespomoćni pred formatima većim od standardnog A4. Sa stanovišta obrade teksta, sa navedenim manama se može živeti, ali kad na red dođe crtanje, jaz između želja i mogućnosti postaje nepremostiv. Laserski štampači i pored preciznog crtanja odmah ispadaju iz konkurencije, jer je maksimalni format koji podržavaju apsolutno nedovoljan za ozbiljan rad. Matrični štampači, koliko god kvalitetni bili, ne mogu da pruže potreban kvalitet otiska – prosečni 24-pinac crta liniju tako što pokušava da pritisne najbližu od svojih iglica na pravo mesto, ali kada je iglica veća od razmaka između dve tačke, rezultat je beznadežno stepenast. Ploteri su priča za sebe – odlično crtaju, ali o bilo kakvoj obradi teksta nema ni govora, a i skupi su.

PRINCIP

Za inženjera ili pisce ovo ne predstavlja problem, ali šta sa korisnicima kojima je potrebno i jedno i drugo, kao što su grafički dizajneri ili projektanti, koji svoje radove prate i odgovarajućim tekstom ili priručnicima? Jedno od rešenja ovakvih problema dolazi iz poznate američke firme Houston Instrument u vidu kombinacije štampača i plotera. Reč je o modelu *JetPro 360 Plus* baziranom na *bubble jet* tehnologiji i širokom valjku. *Bubble jet* tehnologija je zasnovana na mastilu koje se kroz sićušne dizne izbacuje na papir. Baš tako – izbacuje: kad dospe u diznu, mastilo se naglo zagreva, usled čega se obrazuje mehurić koji se brzo širi i prska na papir. Dizna se zagreva u kratkim intervalima preko ugrađenih grejnih elemenata. Svaki impuls ima dvostruku svrhu – mastilo u dizni se zagreva gotovo trenutno, isparava i stvara pritisak potreban za nastajanje mehurića, a po prskanju mehurića se stvara potpritisak koji uvlači sveže mastilo u diznu.

Najvažnija osobina ovakvog sistema je jednostavnost konstrukcije, koja ima tri osnovne prednosti u odnosu na klasične sisteme: povećanu trajnost (čime se smanjuje glava za štampanje), veću gustinu ispisa, odnosno broj „tačkica" (veća rezolucija) i tihi rad.



I ploter i štampač u isti čas: JetPro 360 Plus

PRVI UTISAK

Već samo otvaranje ogromne kutije i vađenja štampača teškog gotovo 13 kg govori nešto o solidnosti izrade, bez obzira što se radi o štampaču sa širokim valjkom. Neobičan ali privlačan dizajn, koji odudara od „kontakcijskog" izgleda, odaje američko poreklo, a precizne, nešto zaobljene ivice i vrlo precizni spojevi pokazuju da je reč o firmi sa imenom. No, da bi se do tle došlo, trebalo ga je prvo otpakovati i ukloniti gomilu osigurača i lepljivih traka, što nije bio ni malo jednostavan posao. Samo sklapanje i instaliranje štampača je bilo prava dečja igra – JetPro 360 Plus je očigledno dizajniran za krajnjeg korisnika koji nema ni volje ni vremena da se bakće sa sporednim stvarima. Potrebno je samo namestiti gornji poklopac (izrađen od solidnog dimljenog pleksija), utaknuti kabl za napajanje, kabl za povezivanje sa računarom, umetnuti papir i štampač je već spreman za rad. Naravno, tu su i DIP prekidači koje treba podesiti po želji, a CAD korisnici (koji očigledno predstavljaju glavnu ciljnu grupu), treba još da instaliraju odgovarajući drajver i hardverski ključ koji se dobija uz štampač.

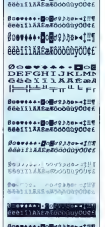
Pogled ispod poklopca otkriva samo izvanredno robustan i precizan mehanizam za vođenje papira, uz nešto neizbežne plastike i mnogo više metala nego što je to uobičajeno. Drugo ništa nije moguće videti, sve je temeljno zatvoreno, uključujući i samu glavu smeštenu na „parking" u levom uglu kućišta. Na prednjoj strani kućišta se nalazi otvor za umetanje pojedinačnih ili nestandardnih listova papira, sa nizom kontrolnih tastera i pripadajućih lampica smeštenih na kontrolnom panelu neposredno iznad otvora. Uz ovaj otvor se nalaze i vratašca koja skrivaju kasetu sa mastilom i koja po konstrukciji neodoljivo podsećaju na poklopce koji na modernim televizorima i video-rikorderima skrivaju kontrolnu dugmad. Na gornjoj ploči je smešten potisni traktor uverljivog izgleda, koji potpuno nestaje sa očiju montažom uređaja za automatsko ubacivanje pojedinačnih listova (CSF – *cut-sheet feeder*). Na zadnjoj ploči se nalaze priključak za napajanje i Centronics port, svaki na po jednom kraju, kako je to i red, i DIP prekidači (treba li reći da su i oni skriveni ispod odgovarajućeg poklopca?). Sve u svemu, vrlo solidna i dopadljiva mehanička celina.

PERFORMANSE

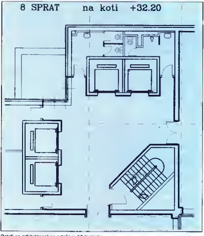
Neugledni dodatak „Plus" u nazivu bi trebalo da označava poboljšanje ili pojačanje osnovnog modela. Međutim, ovde je reč o detaljnoj reviziji prethodnika u vidu potpuno novog kućišta, novog ROM-a, novog seta Escape sekvenci, novoj emulaciji, novim fontovima, većim baferom i još nizom krupnijih i sitnijih izmena. Pored ovih vidljivih, uglavnom kozmetičkih, glavne promene odnose se na povećanje brzine od dobrih 50% – deklarisana brzina sada iznosi 300 cps u tzv. *High Speed* modu (ranije 220) i 150 u HQ (High Quality) modu (ranije 110). Ova brzina, kao i kod svih štampača

predstavlja samo lepa želje konstruktora i kao takva je neostvarljiva u praksi – kvalitet štampača se uglavnom određuje po stepenu odstupanja od te cifre. JetPro Plus je tu pokazao veoma dobre rezultate: u HS modu je postigao 240 cps, u HQ modu punih 120 cps, dok je u testu sa realnim tekstom (dvostruki prored, margine, dosta bolda i kurziva, fusnote) postigao i daje odličnih 103 cps, naravno u HQ modu. Svi ovi podaci zaista impresivno deluju, ali nikakav opis ne može da dočara osećaj koji takva brzina pruža u praksi, budući da je to duplo brže od bilo kog 24-pinskog štampača. Posebno me je fascinirala brzina štampanja grafike – punu A4 stranu bitmapirane grafike je odštampao za manje od minut i po, dok mu je za istu takvu stranu punu vektorske grafike bilo potrebno nešto manje od četiri minuta.

Korisniku koji je navikao na matrične štampače, naročito one brže, štampanje na ovom štampaču izgleda prosto nestvarno – i pored zaista fascinantne brzine, od one dreke tako karakteristične za matrične štampače nema ni govora. Prvi pogled na otisak izaziva isto takvo oduševljenje – kvalitet je skoro laserski, uz je-



Detalj iz automatskog testa (self-test)



Detalj sa arhitektonskog crteža u A2 formatu

dva primetne nejasnoće po kružnim ivicama, za koje sam, uzgred budi rečeno, siguran da će ostati neprimećene od svakog normalnog korisnika.

Grafički otisak je još impresivniji, budući da je ovaj štampač jedan od retkih koji je uspešno prošao i jedan od najtežih testova, a to je korektno štampanje sivog rastera. Sivo je bilo zaista sivo, onako kako je izgledalo i na ekranu, uz primetne razlike u gradacijama. Ovakav otisak sam i očekivao znajući da je reč o glavi sa 48 mlaznica (ekvivalent iglica), rezoluciji od 360 dpi i slovnoj matrici od 36 x 48 tačaka.

Što se fontova tiče, na raspolaganju su tri fonta (*Courier*, *Prestige* i *Gothic*, odnosno sans serif) sa 10, 12, 15, 17 i 20 cpi i propotional. Od efekata su dostupni bold, kurziv, podvučeno, nadvučeno, dvostruko šira, dvostruko više (i kombinacija), povećano za proizvoljan horizontalni i vertikalni faktor, senčeno, inverzno, outline i još dosta drugih. Onima kojima ni to nije dovoljno, na raspolaganju stoje i dva slota za font-kartridže koje mogu da biraju softverski ili sa kontrolnog panela. Pored toga, bafer od 32 KB se sada može proširiti za dodatnih 128 KB, a logična posledica tog koraka je i mogućnost istovremene upotrebe šest softverski definisanih fontova od po 256 karaktera (ranije 1 font sa 128 karaktera). Model JetPro 360 (bez plusa) je raspolagao IBM Proprinter XL emulacijom, dok Plus može da emulira Proprinter XL24E i Epson LQ 1050.

Budući da je reč o printer/ploteru, uz JetPro 360 Plus se dobija i disketa sa programskom podrškom i hardverski ključ koji treba utaknuti između printera i računara. Reč je o posebno razvijenim programima *ProPlot* i *JetView*, kao i o ADI drajveru za *AutoCAD*. Za upotrebu uz *AutoCAD* treba instalirati ADI drajver (default vrednosti i interapti su standardni i objašnjeni u uputstvu) i pokrenuti ga pre *AutoCAD*-a. Plotovanje na štampaču svakako nije stvar u čijoj se brzini može uživati – prvo ADI interpretira *AutoCAD*-ove podatke pa ih zatim šalje štampaču, ceo posao se odvija po sistemu linija po linija, tako da štampanje iole većeg crteža ume da potraje dovoljno za kaficu. Ipak, to je minorna zamerka u odnosu na sve prednosti ovakvog pristupa.

Pored direktnog interpretiranja, moguća je i emulacija – ProPlot direktno emulira kako Houston-ov DM/PL plot jezik, tako i HP-ov HPGL. Program omogućava postavljanja standardnih parametara i rezolucije, kao i izbor odgovarajuće emulacije. *JetView* je zamišljen kao spona za povezivanje sa digitajzerima i skenerima uz podršku RLC i TIFF standarda, uključujući i Houston-ov sopstveni Scanpro sistem za priključivanje na velike plotere. Na taj način je omogućeno čitanje crteža skenerom i njihovo (bit-mapirano) štampanje. Program podržava većinu standardnih displeja, miševa i digitajzera, a dozvoljava i zumiranje, odnosno pravljenje isečaka. Oba programa su koncipirana

na isti način imaju isti *image*. Iako da se korisnik vrlo brzo navikava i prestaje da ih primećuje u svakodnevnom radu.

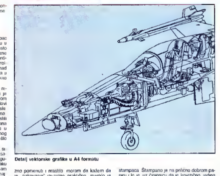
RAD SA PAPIROM

Rad sa papirom je ono što ovaj štampač čini izuzetnim – pojedinačni listovi se ulažu u pokretnu vodilicu na prednjoj ploči koja u isto vreme služi i kao baždareni graničnik za razne formate. Papir se jednostavno namesti i pritiskom na dugme uvuče na mesto. Na kontrolnom panelu koji se nalazi neposredno iznad otvora se može podesiti mikropozicija papira u veoma malim koracima, ali se to pokazalo kao nepotrebno.

Traktor za perforirani papir polisnog tipa nije ništa manje pouzdan i precizan, mada je štampač ove klase grehota koristiti sa papirom takvog kvaliteta. Na kraju, pojedinačni listovi do formata A3 se mogu ubacivati i automatski uz pomoć CSF a koji radi sa jednim ili dva magacina istovremeno, tako da se mogu koristiti dve vrste ili dva formata papira. Sva tri načina papira karakteriše izuzetna preciznost, do u milimetar, a u toku testa nije došlo ni do jednog jedinog gužvanja iako je kroz štampač prošlo preko 350 listova papira raznih formata.

Zasluga za ovo, pored precizne izrade, sigurno pripada i sistemu za vođenje koji se sastoji od dva masivna gumena valjka i niza gumenih pritiskivača papira koji eliminišu svaku mogućnost greške. Sve do ovog testa sam smatrao Epsonove mehanizme ubedljivo najboljim i najsigurnijim, ali sada se pokazalo da i od najboljeg postoji bolje.

Kad je već reč o štampanju, treba obavezno pomenuti i mastilo, moram da kažem da je „pakovanje" izuzetno praktično, mastilo je smešteno u plastičnu kasetu koja se ulaže u predviđeni otvor na prednjoj ploči, cela operacija je jednostavna da jednostavnija ne može biti, a mastilo je pokazalo začuđujuću izdržljivost. Tokom testa sam odštampao preko 300 strana gustog teksta, sve u HQ modu, i oko 50 strana grafike, a da se gustina otiska nije ni najmanja promenila. Ako se ima u vidu da na jednu stranu (A4) prosečno staje oko 4 000 znakova, a da jedna grafička strana potroši mastila, trake ili tonera kao 5 tekstualnih, onda znači da je tokom testa odštampano oko 1 500 000 znakova. Time je deklarisani vek kasete od milion znakova prebačen za 50% bez znakova „zamora"!



Detalj vektorske grafike u A4 formatu

Lična karta:

Dimenzije (ŠxVxD) 610x362x137 mm
Težina 13 kg
Broj dizni u glavi 1 x 48
Vek trajanja glave
- 100 miliona HQ karaktera (MCTF)
- 200 miliona HS karaktera (MCTF)

Bafer 32 KB, proširivo do 128 KB
Papir
- pojedinačno, do A2
- pojedinačno sa doturačem, do A3
- traktor, do A3

Veznici Centronics, RS 232 (Opcija)
Emulacije
- IBM Proprinter XL24E
- Epson LQ 1050
- HPGL (uz priloženi softver)
- DMPL (uz priloženi softver)

Standardni fontovi
- Courier 10, 12, 15, 17, 20 cpi
- Prestige 10, 12, 15, 17, 20 cpi
- Gothic(Sans Serif) 10, 12, 15, 17, 20 cpi

Korisnički definisani fontovi
do 6 fontova od po 256 karaktera
Matrica maksimalno 36 x 48 tačaka
Grafika maksimalno 360 x 360 dpi
Tekstualna brzina (10 cpi, A4)
- Draft deklarisana, 300 cps
- Draft izmerena, 240 cps
- Letter deklarisana, 150 cps
- Letter izmerena 120 cps
- Letter, realni tekst, 103 cps

Grafička brzina (puna A4 strana)
- Bit mapirana slika 1'23"
- Vektorska slika 3'42"

ZAMERKE

Jedini način na koji se ovakvi testovi mogu raditi je traženje zamerki. Ovog puta sam imao veoma težak zadatak, budući da je sve funkcionisalo besprekorno. Ako bi se ipak tražila zamerka, onda treba reći da kvalitet otiska veoma zavisi od vrste i kvaliteta papira koji se koristi, no to svakako ne ide na dušu konstruktorima – to je slaba tačka svih ink-jet, *bubble-jet* i laserskih štampača. Znatno ozbiljnija zamerka, na koju bi konstruktori mogli da utiču, je nepostojanost mastila na otisku. Sa štampanim materijalom treba biti obazriv, jer se mastilo relativno lako razmazuje, a o slučajnom vlaženju papira da i ne govorimo, istina, ovo je manje izraženo nego kod većine srodnih štampača, manje čak i u odnosu na HP DeskJet, ali ipak nije stvar koja se ne bi mogla otkloniti.

Pored ovoga, uspeo sam da pronađem još samo dve slabe tačke – uputstvo i kvalitet komandnih tastera. Komandni tasteri su svoj posao odlično obavili ali su nekako suviše nežni, kao da su pravljeni za žensku ruku. Kružnog su oblika i taman dovoljne veličine da na njih stane jagodica prsta – kao korisnik bih bio mnogo srećniji da su četvrtasti i veće površine. Ako se o ukusima ne raspravlja, uputstvo mogu mirne duše da proglasim najgorim delom štampača. Štampano je na prilično dobrom papiru i to je, uz činjenicu da je trojezično, jedina pozitivna stvar. Ilustracije, posebno one u uvodnom delu posvećenom montaži su dosta traljavo i nepregledno urađene, a uz to su još i toliko male da se gube svi detalji. Priručnik pod naslovom *User's Manual* je bukvalno to i ništa drugo – upustvo za upotrebu. Data su objašnjenja za upotrebe kontrolnog panela, ulaganje papira, montiranje CSF-a i podešavanje DIP prekidača, sve dosta spartanski. Od pregleda Escape sekvenci ni govora, kao ni o finesama za one koji žele da štampač iskoriste do maksimuma. Zaista je prava šteta što proizvođač ovakvog renomea uz ovakav štampač nije obezbedio i upustvo odgovarajućeg kvaliteta. Ovo je zaista prilika za čuvenu rečenicu: Ako može Epson, zašto ne mogu i ostali?

ZAKLJUČAK

Nema prijatnijeg osećaja za korisnika niti težeg zadatka za onoga ko testira od rada sa uređajem kod koga radi sve što treba onako kako treba. Očigledno je reč o promišljenom i uravnoteženom proizvodu namenjenom pre svega korisnicima *AutoCAD*-a, ali i svim ostalim korisnicima kojima su potrebni odličan tekstualni i grafički otisak za tehničku dokumentaciju svake vrste – sa te strane je ovaj štampač pun pogodak. JetPro 360 Plus može da zadovolji sve ukuse i sve zahteve izuzev štampanja višestrukih kopija, a ako se ima u vidu i njegova cena od približno 50 000 dinara sa svim dažbinama i garancijom, onda ga možemo mirne duše preporučiti i svim ostalim korisnicima koji traže kvalitetan otisak ravan laserskom, brzo i tiho štampanje i pouzdan uređaj sposoban za štampanje i na formatima većim od standardnog A4.

Kontakt adresa

KMA
Milentija Popovića 9, Sava Centar
11000 Beograd
Tel. 222 19-28

OTVORI PROZOR DRAGA

Medu malo upućenijima u računarskom svetu postoji jedno staro pravilo: nikada ne treba kupovati programski paket koji nosi oznaku „x.0", niti na njemu treba zasnivati budući sopstveni razvoj, već je bolje sačekati verziju „x.1" ili „x.01" u kojoj će sve greške (ili barem većina) biti ispravljene. Microsoft je uložio velike napore da sa svojim proizvodima *Windows 3.0* i MS-DOS 5.0 ovo uvreženo mišljenje opovrgne.

Ali u Microsoftu nisu posle toga sedeli besposleni. Korak ka zaštićenom režimu – neosporno najvažnija novina u *Windows 3.0* – bio je tek početak. Zaštićeni režim je memoriju, koja bi se mogla upotrebiti u mnoge korisne svrhe učinio raspoloživom, pa je dalji razvoj *Windows*-a bio neminovan. Tako danas pred sobom imamo najnoviju verziju – *Windows 3.1*.

PREKO 1000 FUNKCIJA

Čaršijske priče su već, pretpostavljam, čuli. *Windows 3.1* bi trebalo da bude „*Windows 3.0* sa *TrueType*-om i nešto manjih poboljšanja". Zapravo, *Windows 3.1* donosi dosad najobimnije proširenje API-ja u svojoj šest godina dugoj istoriji. Korisnicima sasvim sigurno neće biti dosadno.

Iz Tabele 1 možete videti da *Windows 3.1* ima za oko jednu trećinu više funkcija nego *Windows 3.0*. Sa proširenjima za multimediju i tzv. „pen" računara broj funkcija premašuje 1000. Zadovoljni? (Imajte na umu dok čitate ovaj tekst da se sve informacije odnose na beta-verziju programa. *Windows* se, dakle, može još ponešto izmeniti.)

Prelazak sa *Windows*-a *2.x* na *Windows 3.0* za jedne je bio lak, a za druge težak. Kada se *Windows 3.0* pojavio, svakom (?!) je bilo jasno da mu nedostaje *TrueType*. *TrueType* je jedna tehnika ispisa firme Apple, koja je za Macintosh i *Windows* obećala WYSIWYG okruženje visokog kvaliteta. Prelaz na *TrueType* imaće za posledicu neke izmene u dosadašnjim programima, o čemu će još biti reči. Sa druge strane, prelazak sa *Windows*-a *3.0* na *Windows 3.1* trebalo bi da bude mnogo lakši nego prelazak sa *Windows*-a *2.x* na *Windows 3.0*, jer ćete odmah želeti da isprobate nove funkcije.

Neke od novih funkcija odično su nam nedostajale već kod verzije 1.0. (Sada postoji standardan dijalog za rad sa podacima i direktorijumima.) *Windows 3.1* takođe uvodi i OLE – skraćenicu koja obećava da postane najčuvenija posle DDE (OLE = *Object Linking and Embedding*, DDE = *Dynamic Data Exchange*). Kao i DDE, tako će i OLE biti standardna stavka u uporednim tabelama karakteristika *Windows* programa.

Jedna od interesantnijih izmena je da se *Windows* više ne izvršava u realnom režimu (real mode). To je urađeno zbog potrebe da se izvršava što više *Windows* programa koji ionako ne rade u realnom režimu, a i većina korisnika paketa *Windows* ima potrebnu konfiguraciju hardvera (80286 ili bolji i otprilike 1 megabajt memorije) da bi *Windows 3.1* mogao da se izvršava i u standardnom režimu.

ŠTA IMA NOVO

Ranije je bilo uobičajeno za *Windows* programere da DOS programe startuju iz DOS komandne linije, a da *Windows* pozivaju samo pri testiranju programa. Ali uz takve programske jezike kao što su *Borland C++*, *Turbo Pascal* za *Windows*, *Microsoft Visual Basic* i *Quick C* za *Windows* – *Windows* programer postaje i korisnik *Windows*-a.

Verzija *3.1 Windows*-a sadrži još mnogo proširenja koja će korisnicima pasti u oči. Najvažnije je novi *File Manager* (mogli biste sada reći – videli smo kako stari radi, a evo ga već novi). U verziji *3.0 File Manager* je jedan MDI program (Multiple Document Interface) koji ili prikazuje samo jedan prozor sa stablom direktorijuma ili dodaje jedan ili više prozora koji prikazuju datoteke poddirektorijuma (slika 1).

U prvoj verziji *File Manager*-a prelazak sa jednog na drugi disk drajv nije bio naročito dobro rešen. Posle svakog prelaska na drugi drajv *File Manager* je morao iznova da formira prikaz stabla direktorijuma. U *File Manager*-u *Windows*-a *3.1* prozori se dele na dve polovine. Leva polovina prikazuje stablo direktorijuma, a desna datoteke iz odabranog direktorijuma (slika 2). Možete prikazati i stabla direktorijuma za više drajvova (otvara se prozor za svaki drajv, što se takođe vidi na slici 2). I što je najbolje, *File Manager* pamti koji su prozori otvoreni. Ako završite rad sa *Windows*-om, pa ga kasnije nastavite od tog mesta, *File Manager* će ponovo prikazati sve ranije otvorene prozore. Ovo je zgodna stvar ako čovek mora često da izlazi iz *Windows*-a.

U verziji *3.1* je *File Manager* i dosta brži nego njegov prethodnik. Starom *File Manager*-u je za formiranje stabla direktorijuma za hard disk na kome je bilo 500 direktorijuma trebalo 35 sekundi. Novom *File Manager*-u je za to sasvim dovoljno 29 sekundi, a kako se čuje, poboljšanje u računarskoj mreži bi trebalo da bude još značajnije.

Program Manager je dobio jednu novu grupu programa pod nazivom *Startup*. Ako neki program zavedemo pod tu grupu, tada će se pri startovanju *Windows*-a startovati i taj program. Ovo je mnogo jednostavnije od komande RUN= u datoteci WIN.INI.

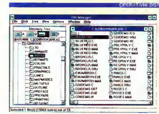
Windows, osim toga, vodi računa i o onom malom programu koji vašem ekranu čini neobične stvari ako dovoljno dugo ne pokrenete miša ili nešto ne otkucate. Protokol za uključivanje ekrana je isti kod *Windows*-a *3.1* kao i kod *Windows*-a sa proširenjem za multimediju.

NEPRIHVATLJIVE API-GREŠKE

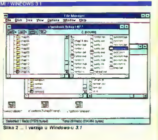
Jedna od najvećih pogodnosti koje *Windows* u zaštićenom režimu ima da ponudi odnosi se na nepopravljive greške u aplikaciji (dalje ćemo koristiti skraćenicu UAE – *Unrecoverable Application Error*). Zaštićeni režim pomaže *Windows*-u da prepozna nedozvoljena posezanja za memorijom i da za to odgovorni program prekine (to je najmanje što očekujemo) pre nego što on u našem računaru napravi haos. A jedna od najgorih stvari sa kojom *Windows* mora da se nosi u zaštićenom režimu je UAE. Međutim, kada do greške dođe, nema informacije ko ju je prouzrokovao. Program? Neka DLL? Neki nevažeći poziv funkcija? Ili neka pogrešna funkcija? A nemate nikakvu mogućnost da tu grešku ispravite. Isto tako, kad se program završi, nije sigurno da li je *Windows* iza njega sve dobro „počistio". Možda je *Windows* još u nekom nestabilnom stanju, ili se jednostavno „sruši".

UAE se ubraja među najomraženije „karakteristike" *Windows*-a *3.0*, što je zahtevalo hitne prepravke. Ali to je lakše bilo reći nego učiniti. Način rada u zaštićenom režimu uveden u *Windows 3.0*, pa čak i u *Windows 3.1*, na isko-

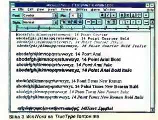




Slika 1 *File Manager* u *Windows-u 3.0*



Slika 2 ... I verzija u *Windows-u 3.1*



Slika 3 *WinWord* sa *TrueType* fontovima

iščava mogućnosti procesora 80286 i 80386 da „čisto" odvajaju procese. Svi programi, pa i sam *Windows*, međusobno dele istu lokalnu opisnu tabelu (LDT).

Microsoftova istraživanja su pokazala da 40% UAE-a nastaje zbog samih programa i zbog grešaka u adresiranju, dok ostalih 60% za razlog ima nevažeće argumente API-funkcija. Na primer, jedan misteriozni UAE javlja se i u *Windows*-u *3.1* (kažu da će biti otklonjen) pri pozivu funkcije **PolyPolygon** sa 1 kao poslednjim argumentom. Najbolje bi bilo kada bi API funkcije temeljno proveravale da li su pristigli argumenti važeći – bile bi „vodootporne" kako se samo poželeti može. Tada bi eventualni UAE bio samo jedna „neprihvatljiva API-greška" (Unacceptable API Error).

Microsoft je pokušao da u verziji *3.1* boljom proverom argumenata smanji broj UAE-a. Osim toga, izmenjen je i prozor sa porukom o UAE. U verziji *3.0* jednostavno piše

UNRECOVERABLE APPLICATION ERROR
Terminating current Application.

Kod *Windows*-a *3.1* ovaj prozor pruža dodatne informacije, kako bi onaj ko razvija program lakše mogao odrediti uzrok greške.

Application Error
BADAPP caused a General Protection Fault in module BADAPP.EXE at 0001:00AE
Press Close. BADAPP will close

Proizvod	Izdat	Broj funkcija
Windows 1.0	novembar 1985	379
Windows 2.0	novembar 1987	458
Windows 3.0	maj 1990	578
Windows 3.1	–	771
Multimedia	–	197
Windows for pens	–	78

Ne zna se da li će ovakav prozor ostati i u komercijalnoj verziji *Windows*-a *3.1*, ali u beta verziji postoji jedan simpatičan mali program po imenu *Dr. Watson* koji daje dodatne informacije ako se pojavi UAE. *Dr. Watson* formira jednu LOG datoteku i u nju upisuje adresu vinovnika, par disasembliranih naredbi do mesta te greške, sadržaj CPU registara, trenutni stek, listu procesa učitanih DLL-ova, kao i par informacija o stanju samog *Windows*-a. Ljudi koji su testirali verziju *3.1* tvrde da su uz pomoć *Dr. Watsona* pronalazili UAE u svojim programima koji su ih mučili mesecima!

NOVI STANDARD ZA FONTOVE

Definicija pojma WYSIWYG se, što se tiče tekst procesora, sigurno izmenila za poslednjih 10 godina. Da li biste poverovali kada bi vam neko danas rekao da je *WordStar 3.x* jednom važio za WYSIWYG program samo zato što je tačno pokazivao u kom se redu i na kojoj stranici nalazite dok kucate?! Mnogi tekst procesori tog vremena nisu imali tu pogodnost. Kasnije će *Microsoft Word* biti smatran WYSIWYG-om jer može na ekranu da prikaže podvučena slova, kao i **bold**, *kurziv*. Kasnije je došao *Windows Write* koji je WYSIWYG zato što može na ekranu prikazati razne fontove i to u različitim veličinama i stilovima.

Mada *Windows* raspolaže sa više fontova u raznim veličinama, ipak se formatirani prikaz teksta na ekranu nikada u potpunosti ne slaže sa odštampanom stranicom. Problem leži u tome što *Windows* ekranske fontove memoriše kao rasterske (bit-map) slike. Tako su formirana slova različitih stilova i veličina koja se teško mogu promeniti. Rasterski font može biti povećan samo za neki celobrojni faktor, pri čemu će redovi i kolone matrica slova biti prikazani više puta, ali to dovodi do upadljivo „stepenastih" slova. Jedina alternativa rasterskim fontovima bili su vektorski fontovi. Kako su vektorski fontovi sastavljeni od pojedinačnih linija, moguće ih je bez problema povećavati ili umanjivati što je prava blagodet za štampače.

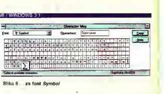
U međuvremenu su i štampači znatno poboljšani. Na primer, PostScript doneo fleksibilnost glatkih i nenazubljenih konturnih (outline) fontova. Ova slova se mogu povećavati, iskositi, rotirati, a da ne izgube svoj oblik. Ko je od programera radio sa GPI (Graphics Programming Interface) u OS/2 1.1, mogao je steći utisak o mogućnostima konturnih fontova.

Konturni fontovi **GPI**-a imaju jednu vrlo krupnu manu. Njihovo skaliranje se zasniva na jednoj utvrđenoj šemi, bez obzira na tipografsku ispravnost. Na primer, usled greške pri zaokruživanju, dva kraka slova H u fontu helvetika biće različito udaljena, a nekad i samo za širinu 1 piksela na ekranu. Ova netačnost pri uobičajenim veličinama slova na ekranima odvodi od praktično nečitljivih tekstova. Ispitivanje tipografske ispravnosti u toku podešavanja veličine naziva se na engleskom „hinting". Ali, odk treća strana (ostali proizvođači) nije izašla na tržište sa svojim outline-fontovima kao što je Adobe nastupio sa svojim *Type Manager*-om, *Windows 3.0* je morao da se pojavi sa fontovima bez ovih presudnih „saveta" za korekturu.

Windows 3.1 uvodi jedan jedini postupak pod nazivom *TrueType* za outline fontove. *TrueType* je prvobitno razvijen u Apple-u za Macintosh, a takođe je ugrađen i u novi System 7.0, dok je Microsoftu prodala licenca. Tako se *TrueType* našao na dve najrasprostranjenije linije personalnih računara, sa dobrim



Slika 4 Tabela rasporeda znakove za font *Arial* i



Slika 5 za font *Symbol*

šansama da jednom postane i industrijski standard. *TrueType* je otvoreni standard – svako ko razvija fontove može ih oblikovati u *TrueType* formatu. *TrueType* fontovi mogu biti prikazeni na svakom monitoru i grafičkoj rezoluciji koju *Windows* podržava.

Microsoft isporučuje *Windows 3.1* sa *TrueType* fontovima (slika 3) koji predstavljaju pandan standardnim PostScript fontovima (*Courier, Helvetica* i *Times Roman* u normalnom, bold, kurziv i bold-kurziv stilu, kao i starogrčki alfabet). Za nazive *Helvetica* i *Times Roman* Microsoft nije uzeo licencu, pa ćemo se morati navići na nazive *Arial* i *Times New Roman* (*Courier* ostaje *Courier*). Ovih 13 *TrueType* fontova izgledaju isto kao i odgovarajući PostScript fontovi.

Svaki od ovih 13 fontova je smešten u svoju posebnu datoteku sa ekstenzijom TTF (True Type Font). Veličine ovih datoteka se krecu između 53 K i 70 K. Ovo su prilično velike datoteke u poređenju sa onima za rasterske fontove, ali mora se imati na umu da jedna jedina *TrueType* datoteka sadrži podatke o slovima potrebne za sve izlazne uređaje, sve grafičke rezolucije i sve veličine slova.

Microsoft preporučuje da se za prikaz teksta na ekranu i štampaču odsad koristi samo *TrueType*. Svakako će korisnici *Windows*-a mogu isključiti *TrueType* fontove ili ih cak i izbrisati. Da bi budući programi bili upotrebljivi za svakoga, treba ih uputiti na *TrueType* fontove, ali uprkos tome ne bi trebalo zapostaviti ni stare rasterske i vektorske fontove.

Ukoliko program zna da na raspolaganju ima *TrueType* fontove, onda je prelazak na njih vrlo jednostavan. Polje **lfFaceName** iz strukture podataka LOGFONT se definiše kao **Courier**, **Arial**, **Times New Roman** ili **Symbol**. Osim toga, u polje **lfHeight** se postavi željena veličina fonta u logickim jedinicama. Zatim se prema potrebi polje **lfWeight** postavi na 700 (bold), a **lfItalic**, **lfUnderline** ili **lfStrikeOut** na 1. Ukoliko želimo da imamo slova posebne širine, možemo polje **lfWidth**, koje je na nuli, postaviti na željenu vrednost. Sva druga polja iz LOGFONT strukture mogu biti popunjena nulama.

Za ispis na ekran slova konturnog fonta biće preračunata na željenu visinu, a zatim će kao rasterske slike biti smeštena u memoriji. Kako se gotove rasterske slike nalaze u memoriji, ispisuju se jednako brzo kao i obični rasterski fontovi. Kada se *TrueType* font šalje na štampač, GDI šalje znake ili u obliku rasterskih slika, ili kao ispunjene poligone, ili pak drajver konvertuje *TrueType* u neki oblik koji će se moći instalirati na štampaču.

Kurzivna slova u *TrueType*-u su zaista kurziv, a ne prosto „naherane" verzije normalnih slova. Ovo je primetno u razlici malih slova „a" i „f" između fontova *Times New Roman* i *Times New Roman Italic* (slika 3). Nakoliko ranije neiskorišćenih ASCII kodova od 128 do 159 decimalno biće sada popunjeni sa nekim cesto potrebnim znacima (slika 4). Font *Symbol* je kompatibilan sa odgovarajućom PostScript verzijom (slika 5).

Postojeći *Windows* programi koji rade sa različitim fontovima uglavnom pozivaju **EnumFonts** i saznaju iz ovog pokazivača koje fontove mogu ponuditi korisniku. Neki od ovih programa će moći i bez izmena da rade sa *TrueType*-om, a neki neće. Program koji ne radi sa vektorskim fontovima neće raditi ni sa *TrueType* fontovima.

Zbog činjenice da stari rasterski fontovi koriste ANSI set karaktera, a da stari vektorski fontovi koriste OEM set karaktera, dakle IBM set, mnogi programi uzimaju da se samo fontovi sa OEM setom karaktera mogu skalirati. Što nije ispravno. Zato se programski kod mora promeniti ako bude potrebno uvođenje *TrueType*-a. *Windows* program koji korisniku omogućava da bira između svih vektorskih fontova, a takođe mu omogućava i precizno postavljanje veličine slova – odlična je osnova. Kod njega će verovatno bez daljih izmene biti moguće uvođenje *TrueType*-a.

Da li je reč o *TrueType*-u može se programski utvrditi iz novog flega SCALABLE_FONTTYPE u parametru **nFontType** funkcije **EnumFonts**. Ako je taj bit setovan, onda je pismo *TrueType*. Druga metoda da se prepozna *TrueType* font je poziv funkcije **GetTextMetrics**. Niža 4 bita polja **lfPitchAndFamily** sadrže iscrpne informacije. Bit 2 je novi *TrueType* bit. Ako program treba da koristi isključivo *TrueType* fontove za ekran i štampač, onda on jednostavno ignoriše sve fontove kod kojih fleg SCALABLE_FONTTYPE nije setovan. Isključiva upotreba *TrueType*-a rešava neke osnovne probleme. Prvo, slova na ekranu i štampaču biće jednaka i po izgledu i po veličini. Drugo, i ekran i štampač imaju isti set karaktera. I treće, dokumenti izrađeni pomoću *TrueType*-a na PC-u biće vrlo slični dokumentima izrađenim na Macintosh-u sa *TrueType*-om. Međutim, isključivim favorizovanjem *TrueType* fontova mogli bi se naljutiti oni korisnici koji su svoj novac uložili u font-kertridže i datoteke. Bolje rešenje je omogućiti širok izbor svih raspoloživih pisama, zajedno sa *TrueType*-om, pa ko šta voli.

Verovatno će programeri napisati neki prozor za dijalog, u kome će korisniku omogućiti da izabere željeni font sa listom svih fontova i standardnih veličina, kao i mogućnost da tačno unese veličinu za neki skalabilni font. (Ako mislite da nije zabavno programirati dijalog prozore, sačekajte još malo, a tada stižu dobre vesti.)

Odlična stvar kod konturnih fontova je što vam omogućavaju da sa njima radite kao sa svim ostalim grafičkim objektima. Slovo možete, na primer, vrlo lako obrtati. Microsoft se svakako pri ugradnji *TrueType*-a u *Windows 3.1* koncentrisao na brzinu i čitljivost, a ne na posebne efekte. U prvoj beta verziji *Windows*-a *3.1 TrueType* fontovi nisu reagovali na polja **lfEscapement** i **lfOrientation** strukture LOGFONT, što još može biti promenjeno do pojavljivanja finalne verzije na tržištu.

Windows 3.1 još uvek podržava i stare rasterske i vektorske fontove. Ali rasterski fontovi su dobili nova imena. Stari „Helv" font sada se zove „MS Sans Serif", a stari „TmsRmn" se zove „MS Serif". Ako Vaš program za pozivanje funkcija **CreateFont** i **CreateFontIndirect** koristi stara imena, nemojte oko toga brinuti. Stara imena biće pretvorena u odgovarajuća nova. Ali ako vaš program poziva **EnumFonts** i traži stara imena, onda ćete odgovarajući deo koda morati da prilagodite.

Za ugodniji rad sa *TrueType* fontovima ima nekoliko novih funkcija. **GetRasterizerCaps** daje podatke da li je *TrueType* omogućen i da li je barem neko *TrueType* pismo instalirano. Može se, naime, desiti da je, iako je *TrueType* omogućen, korisnik obrisao sve *TrueType* fontove. Širina znaka može se još uvek pronaći sa **GetCharWidth**. U *Windows*-u *3.1* funkcija **GetCharABCWidths** daje širinu znaka iz nekog *TrueType* pisma u 3 dela. Širine iz A i C su širine praznog prostora ispred i iza znaka, dok je B prava širina znaka.

Nova funkcija **GetOutlineTextMetrics** odgovara staroj **GetTextMetrics**, ali radi samo sa *TrueType* fontovima i ne koristi strukturu TEXTMETRIC već OUTLINETEXTMETRIC. U ovoj strukturi podataka stoje dodatne detaljne informacije o fontu. Funkcija **EnumFontFamilies** radi slično kao i funkcija *EnumFonts*. Za jednu određenu familiju fontova (npr. *ITC Barcelona*) *EnumFontFamilies* daje sva imena fontova (npr. *ITC Barcelona Book*).

Problem kod GDI-a je što nema nijedne funkcije za izradu *TrueType* fontova, kao što to imaju PostScript i GPI. Kao delimičnu zamenu za ovu slabost Microsoft nudi funkciju **GetGlyphOutline**. Ova funkcija vraća skicu željenog znaka. Ta skica se može tipografski korigovati, a i ne mora. Znak će biti isporučen u formi rasterske slike ili poligona, a osim toga možete zadati i matricu rotacije. Ovo sve zahteva malo truda i nije tako lepo kako bi moglo biti, ali funkcioniše u slučaju da vam je potrebno.

I na kraju, imamo još jednu novu tekst-funkciju koja radi i sa *TrueType* i sa starijim fontovima. To je funkcija **GetTextExtentEx**, jedna proširena varijanta funkcije **GetTextExtent**. **GetTextExtentEx** na svom ulazu očekuje niz znakova i jedan podatak o širini, a na svom izlazu daje broj znakova koji se uklapa u zadatu širinu.

JEDNOSTAVNIJA RAZMENA PODATAKA

Skraćenica DDE znači *Dynamic Data Exchange* (dinamičko razmenjivanje podataka). DDE koristi normalne informacione puteve u *Windows*-u i kroz njih omogućava komunikaci-

ju između različitih procesa. Nikome ne pada na pamet da porekne značaj i mogućnosti DDE-a u razmeni podataka među programima, ali rad sa DDE-porukama i razna pravila za baratanje memorijom su ipak stvarno komplikovane.

Windows 3.1 sa svojom bibliotekom DDEML (*Dynamic Data Exchange Management Library*) nudi delimično rešenje ovog problema. To je zbirka funkcija koja se nalaze u starom događajima upravljanom (*event-driven*) DDE-segmentu. *Event-driven* DDE je nastavno i dalje pristupačan, a programi koji ga koriste mogu „razgovarati" sa DDEML programima. Svakako, DDEML funkcioniše samo u zaštićenom režimu.

Iako DDEML obećava da DDE programiranje učini lakšim i pouzdanijim, jedna stvar kod njega može čoveka naviklog na pogodnosti starog DDE-a ozbiljno da obeshrabri. DDEML se sastoji iz 26 novih funkcijskih poziva, 9 novih struktura podataka i 142 nove definisane konstante.

Veliki deo DDEML terminologije vodi poreklo od *event-driven* DDE-a. DDE veza postoji i dalje između servera (programa koji ima podatke) i klijenta (programa koji želi podatke). Drugi deo označavanja je nov. Na primer, staro DDE povezivanje počiva na tri dela: **Application Name** (ime aplikacije), **Topic** (tema) i **Item** (predmet). Kod DDEML-a se sad to zova **Service** (usluga), **Topic** i **Item**.

DDEML izgleda kao skup funkcija koje jednostavno u sebi kriju stari DDE, ali on je više od toga. Na primer, DDEML može napraviti više veza jednu pored druge i dodeliti im uslužne poslove koje neki server ima da ponudi, što smanjuje razmenu podataka po pojedinim vezama.

POVEZIVANJE OBJEKATA I UPOTREBA

Kako bismo na naš jezik ovo preveli? Na primer „povezivanje objekata i ugradnja (ili upotreba)". Kao primer, na engleskom „embedded computer system" znači svaki sistem koji u sebi sadrži i neki računar, ali čija glavna funkcija (sistema) nije ista kao računareva (npr. sistem za navođenje raketa ili miroljubivije sistem za kontrolu medicinskih uređaja).

OLE obuhvata 64 nova funkcijska poziva i bez sumnje je to najveće i najkomplikovanije proširenje u *Windows*-u 3.1 (mada su se Španci toga još davno, davno setili...). OLE pomera težište rada korisnika sa programa na finalni rezultat (nazovimo ga dokumentom) i definiše jedan format za taj dokument. Dokumentom mogu biti obuhvaćeni različiti formati podataka koje razni programi mogu tu upisati ili ih odatle pročitati. Tako se svaki program usredsređuje na posao koji najbolje obavlja, a korisnici mogu kombinovati razne programe u pravljenju svog dokumenta. OLE je verovatno najlakše razumeti kroz jedno poređenje sa ostala dva postupka pomoću kojih *Windows* programi razmenjuju podatke.

U prvom postupku jedan program može dobiti podatke od drugog programa u nekom standardnom formatu, uglavnom kao ASCII tekst, rastersku sliku ili metasliku. Pri tome se samo prenose podaci. Ne postoji nikakva veza sa programom koji je kreirao te podatke (npr. kad se sa diska učitava datoteka drugog programa).

Dalje, neki program može korišćenjem DDE-a dobiti podatke iz drugog programa, koji su i ovde takođe u standardnom formatu (tekst, rasterske slike ili metaslike). Ali ovde klijent može uspostaviti vezu sa serverom koji isporučuje podatke. To funkcioniše do te mere da se izmene u podacima kod servera mogu preneti do klijenta.

I OLE takođe omogućava jednom programu da primi podatke od nekog drugog programa. Ali ovi podaci su prisutni u dva različita formata. Jedan format biće razumljiv samo programu koji ih šalje, dok je drugi format za prikaz (ASCII, rasterska slika, metaslika) koji koristi primalac koji te podatke treba da prikaže na ekranu.

Klasičan primer je neki proračun ili tabela (npr. iz programa za unakrsna izračunavanja) koju treba ubaciti u neki tekst. Ako tu tabelu iskopiramo nekako sa diska (prvi način), tada će od nje postati jedan statički blok teksta koji je izgubio svaku vezu sa svojim poreklom. Posredstvom DDE-a (drugi način) primalac dobija tu tabelu u tekstualnom formatu, ali veza sa pošiljaocem ostaje. Ako startujete program iz koga ta tabela potiče i ako se neke brojke izmene, tada se takođe menjaju i brojke u tekstu tekst procesora.

Sa OLE-om tekst procesor dobija tu tabelu u dva formata – jednom originalnu tabelu u formatu koji prepoznaje samo program pošiljalac, a zatim tekstualnu verziju pogodnu za prikaz. Ako u tekst procesoru hoćete da radite sa tom tabelom, tada će se startovati pripadajući program (kome ta tabela izvorno pripada) i primiće od tekst procesora tu tabelu, tako da je sad možete menjati u njenom izvornom programu. Kad završite taj posao i okončate program, automatski se vraćate nazad u tekst procesor, gde je već smeštena izmenjena tabela.

Mogućnosti OLE tehnike pokazuju se na jednom još boljem primeru. Uzmimo da želite čitaocu koji čita neki vaš tekst na nekom mestu bukvalno da **kažete** nekoliko reči o vašem tekstu. Imate, na primer, program, nazovimo ga *Recorder*, uz pomoć koga možete, kao kasetofonom, nešto snimiti i ponovo reprodukovati. Vaš tekst procesor, naravno, ne razume govor. Zato vi svoj komentar snimate *Recorder*-om i to šaljete tekst procesoru. On memoriše digitalizovani govor ponovo u dva formata – kao digitalizovane reči koje uopšte ne prepoznaje i u formatu za prikaz na ekranu (recimo, da je to slika na kojoj je par usana). U tekstu se prikazuje ta slika. Kada čitalac dođe do te slike u tekstu, uključiće se *Recorder* koji se verovatno nijednom neće prikazati na ekranu. *Recorder* od tekst procesora prima tonove i reprodukuje ih.

Ovim smo samo zagrebali po površini OLE-a, ali se već sada može reći da je to jedan kompleksan sistem sa gomilom novih zamisli.

DIJALOZI BEZ PROBLEMA

Delovima programa sa kojima korisnici svakodnevno rade pripadaju i prozori sa dijalogom za otvaranje neke datoteke (dalje ćemo ih zvati samo „dijalozi"). Korisnik zna da pritiskom na <ALT><D><F> uvek otvara dijalog u kome može birati datoteke sa kojima želi da radi radi.

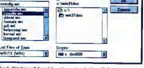
Za programera je pisanje ovakvih dijaloga skoro isto kao i progon u doba Inkvizicije. U retko kojoj drugoj vrsti dijaloga je potrebno tako mnogo posla kao kod ovih za otvaranje datoteka. Većina programera je došla do razumljivog zaključka da su ovi dijalozi za otvaranje datoteka potrebni dovoljno često da bi se isplatilo olakšati njihovu upotrebu u programima.

Rečeno – učinjeno! Nova biblioteka COMMDLG.DLL nudi devet funkcija koje će programerima znatno uštedeti vreme. Tu nije samo funkcija **GetOpenFileName**, koja otvara dijalog za učitavanje datoteka (slika 6), već je tu takođe još i **GetSaveFileName** za snimanje datoteka. I kod štampanja će vam uvek ponovo trebati isti prozor sa dijalogom, i zato COMMDLG sadrži i **PrintDlg** funkciju.

Da bi se korisniku olakšalo podešavanje boja i izbor fontova, COMMDLG sadrži još dva prozora za dijalog – **ChooseColor** i **ChooseFont**. **ChooseColor** dopušta korisniku izbor i promenu boja i to ili po RGB modelu (**Red, Green, Blue**) ili po HLS modelu (**Hue, Lightness, Saturation** – boja, osvetljenost, zasićenje). Dijalog **ChooseFont** pravi razliku između *TrueType* fontova i drugih fontova, a može se ograničiti i samo na *TrueType* fontove (slika 7). Osim toga, za obradu teksta se u COMMDLG-u nalaze i dijalozi **FindText** i **ReplaceText**.

ŠTAMPANJE BEZ *ESCAPE*

Windows funkcija **Escape** još uvek izgleda malo neobično. Upotrebljava se obično kada se žele iskoristiti neke posebne mogućnosti



Slika 6 *Windows 3.1* sadrži gotove standardne dijaloge – primer za *FileOpen*



Slika 7 Spremno za upotrebu dijalog za izbor fontova

NOVI ŽIVOT VAŠE MAŠINE

"Microsoft" *Windows* grafički operativni sistem predstavljen je javnosti u proleće 1990. godine. Reakcija „Byte"-a bila je euforična: „Korisnici DOS-a, veselite se! *Windows 3.0* udahnuće novi život u vaš računar". Ništa, međutim, nije konačno, a najmanje jedan softverski proizvod. Šta možemo da očekujemo od nove verzije popularnih *Windows*-a?

Zoran Kehler

Bez obzira na entuzijazam onih koji su prvi videli novo operativno okruženje „Microsoft"-u za stvaranje *Windows* paketa mora odati dužno priznanje. Paket je napisan tako da maksimalno iskoristi postojeću bazu mašina u kojoj je još uvek preovladivao sa procesorom 80286. Vrlo brzo počela je prava poplava aplikacija pisanih posebno za *Windows*, ili prepravljenih za rad u novom okruženju.

ZVEZDA JE ROĐENA

Pogledamo li malo dublje ispod prijatne spoljašnjosti, još uvek se krije vremešni MS DOS. Ni traga od mogućnosti koje nudi OS/2 32-bitni model programiranja, *multithreaded* arhitektura, poboljšan sistem datoteka i moćan interfejs za grafičko programiranje. Većina aplikacija pod *Windows* sistemom još uvek ne koristi mogućnosti 32-bitnih procesora niti multitaskinga, još uvek su ograničena sporim pristupom tvrdom disku, mogućnostima DOS-a i većinom brigom da se bez većih problema slažu sa velikim brojem rezidentnih rutina i programa za DOS. I pored toga, „Byte" smatra *Windows 3.0* velikim uspehom: „Nakon godina lutanja, „Microsoft" je konačno uspeo da ostvari pogodak sa ovim proizvodom. Probajte ga. Svideće vam se."

Početno oduševljanje vrlo brzo je ustupilo mesto ravnodušnosti, ponegde i kritici programa. Bilo je jako primamljivo instalirati *Windows*, diviti se grafičkim mogućnostima, isprobati onih nekoliko aplikacija koje „Microsoft" isporučuje sa kompletom. Neki su, naravno, otkrili da je igra pasijansa u *Windows*-u jako dobra. U pismu „PC Magazine"-u Michael D. Lore iz Teksasa kaže: „Microsoft *Windows* jeste lep, ali ima nezgodan karakter (...) Ali gledajte, za 35 dolara koliko me je koštao, teško da sam mogao naći bolji pasijans." Svakako da *Windows* ima još mnogo dobrih strana osim simulacije igre kartama, ali i nedostaci ulaze u sliku ovog programa.

"Microsoft" je program napisao tako da do krajnjih granica iskoristi mogućnosti računara.

Međutim, to je za posledicu imalo i to da je *Windows* jako zahtevan u pogledu memorije, brzine procesora i tvrdog diska, kao i grafičke kartice. Istina je da se *Windows* može izvršavati i na mašini sa 286 procesorom sa taktom 10MHz, kao i na grafičkoj kartici kompatibilnoj sa „Hercules" standardom. Istina je, takođe, da će raditi već u 1MB operativne memorije i da neće odbiti da radi sa sporim tvrdim diskom. Rad sa takvom kombinacijom komponenti će, međutim, vrlo temeljito testirati izdržljivost korisnika: program će se sporo učitavati, iscrtavanje ili osvežavanje ekrana traje beskonačno dugo, sve moćnije aplikacije zahtevaju isuviše mnogo memorije, pa je prebacivanje delova iz memorije na disk (swapping) vrlo često i nedopustivo sporo, otvaranje DOS prozora nije tako jednostavno kao što se može pomisliti, a promena drajvera za razne VGA karte, kao i instalacija fontova predstavljaju uvek pravu avanturu. Rešenje hardverskih problema sa *Windows*-om je prosto bezobrazno jednostavno: za rad sa programom gotovo da je neophodan računar sa procesorom 386 na bar 25MHz, a poslužiće i 386SX na bar 20MHz.

Potrebe za snažnijim mašinama nisu, na sreću, odvratile korisnike od upotrebe *Windows*-a. Pozitivne osobine ipak preovlađuju, što se vidi iz sve većeg broja programa pisanih za ovu okolinu, pojave programerskih alata za *Windows*, a takođe i po računarskim sistemima kakve prodaju sve zapadne kuće: „Microsoft" *Windows* je, uz MS DOS, obavezno uključen u cenu.

PRVI UTISCI

"Microsoft" se nada da će nova verzija *Windows*-a ispraviti greške predaka. Beta kopija *Windows*-a *3.1* opravdava ta očekivanja. Reč je o verziji *3.1.034*, Beta Release 1, sa serijskim brojem koji odaje da se stvarno radi o prvim primercima koji su našli put do testera.

Program je došao na 5 disketa od 3.5", formata 1.44MB. Instalacija se neznatno razlikuje od one *Windows*-a *3.0*, iako je vidljivo da su neki od programa za instalaciju promenjeni. Proces je ostao spor, što je i razumljivo ako se uzme u obzir veličina paketa i pokušaj „Microsoft"-a da njen dobar deo automatizuje, time što „prepoznaje" konfiguraciju računara na kome će se *Windows* izvršavati.

Prva novost očekuje vas odmah posle startovanja *Windows*-a: logo je izmenjen i sada je to „Microsoft Windows 3.1". Da se radi o Beta kopiji upozorava natpis „Beta Release 1". Odmah se može primetiti vrlo važna činjenica: nova verzija je značajno brža. Startovanje i inicijalizacija programa mnogo kraće traju. Izgled ekrana **Program Manager**-a je poboljšan: ikone su lepše i jasnije, automatsko razmeštanje ikona bolje je rešeno. Dugački nazivi programa sada se ne preklapaju sa nazivima susednih ikone, već se ispisuju u više redova. Iskustvo u radu sa *Windows*-om naučilo nas je da odmah pogledamo **Control Panel** i mogućnosti podešavanja okoline.

International sekcija bila je prva koju smo pogledali. Želja je bila da se vidi da li je ugrađene podrške za jugoslovenski latinični skup znakova. Od početka razvoja *Windows*-a *3.1* bilo je govora da će paket od MS DOS-a 5.0 naslediti podršku za više kodnih strana, a među njima i za kodnu stranu 852 koja odgovara našem jeziku. Na žalost, beta verzija programa nas nije obradovala tom novošću. U **Country Language** podmeniju nema naše zemlje, ali je primetno da ovaj deo programa nije završen: postoje dve „Other Country" opcije, kao i sedam nedefinisanih jezika.

U **Control Panel**-u ima još novosti. **Fonts**, program za upravljanje ekranskim fontovima, obogaćen je *TrueType* sistemom i novim fontovima. O podešavanju serijskih portova brine se program **Ports**, koji je dobio nove mogućnosti: *Windows* poznaje četiri serijska porta, i za svaki se mogu podešavati svi parametri komunikacije. Novost je da se mogu menjati adrese portova i interapti koji servisiraju portove. **Desk**-

štampača kojima se ne može prići iz *Windows*-a. Ali **Escape** je potrebna i za sasvim normalnu štampu da bi se, recimo, štampanje nekog dokumenta započelo stranice pomerale štampač i završilo. Ali kod upotrebe funkcije **Escape** postoji problem sa formatom parametara.

Windows 3.1 ima šest funkcija za štampanje bez **Escape** i nekoliko novih štampačkih funkcija. Funkcije **StartDoc**, **SetAbortProc**, **AbortDoc** i **EndDoc** zamenjuju odgovarajuće **Escape** funkcije. Umesto **Escape** poziva NEWFRAME i BANDINFO mogu se upotrebljavati funkcije **StartPage** i **EndPage**. Stare **Escape** funkcije su, naravno, zadržane da bi se sačuvala kompatibilnost sa ranijim programima.

Od novih funkcija tu je **ResetDC**. Nju možete pozvati samo između dve štampane stranice, dakle ne između **StartPage** i **EndPage**, i u toku štampe izvršiti neka podešavanja koja ranije nije bilo moguće izvesti.

Prošireni GDI u 32-bitnoj verziji API-a učiniće mnoge od ovih funkcija suvišnim. *Windows 3.1* je napravio prvi korak u tom pravcu, jer **Escape** funkcije za obične poslove sa štampačem nisu više potrebne.

U UNUTRAŠNJOSTI SVETILIŠTA

Pod ovim naslovom ćemo samo ukratko izložiti još jednu novinu *Windows*-a *3.1*. Ako razvijate „savršen" korisnički program, tada verovatno želite i da proverite njegovo ponašanje u „stresnim" situacijama, na primer kada ima malo ili nimalo memorije, kad je hard disk pun i slično. Nova biblioteka STRESS.DLL može vam proizvesti te granične uslove. Iako da možete ustanoviti kako vaš program reaguje. O ovome samo toliko, jer STRESS.DLL neće biti deo standardnog *Windows 3.1* paketa, već će se isporučivati zajedno sa *Windows 3.1 SDK* (*Software Development Kit*).

Šta se sada zaista dešava u *Windows*-u? Da li vam se pri pomisli na to koža naježi, da li vam je čas vrućina, čas hladno, a prsti počinju da vam se tresu? (Dobro, možda malo preterujem, ali tako bi bilo u dobrom horor filmu.) Do sada je programer, kada je želeo nešto da sazna u vezi internih podataka *Windows*-a, morao da posegne za nedokumentovanim postupcima da bi došao do traženih informacija. Nova biblioteka TOOLHELP.DLL nudi 38 funkcija sa kojima programeri mogu tačnije pratiti zbivanja u *Windows*-u i imati manja teškoća pri razvoju dibagera.

TOOLHELP daje informacije o svim tekućim procesima i svim aktivnim modulima, o celim prijavljenim klasama prozora, o svim globalnim memorijskim blokovima. Za primenu u dibagerima obezbeđeno je prebacivanje između procesa, preusmeravanje ulaza i rukovanje prekidima i izuzecima.

ktop je odbio **Screen Saver** program koji zatamnjuje ekran u vreme neaktivnosti Windows-a. Očigledno je da nije dovršen jer neke od opcija za zatamnjivanje pale od grešaka. **Printers** program je dosta poboljšan. Značajnije su promenjeni programi za kontrolu „epson" matričnih štampača i „Hewlett Packard" laserskih štampača. Setup za HP štampače sada koristiše **Page Protection** i način štampanja lokalne sivog.

VAŽNI NOVITETI

O novitetima *Windows*-a *3.1* se detaljno govori u članku „Otvori prozor, draga" koji su napisali Microsoft programeri, pa ćemo ovde izneti samo svoje najosnovnije utiske.

O *TrueType* tehnologiji gotovo se sva znalo još pre pojave *Windows*-a *3.1*. Radi se o načinu generisanja konturnih fontova koji je razvijen u „Apple" kompaniji. Fontovi se skaliraju, pa prema tome ne bi trebalo da bude dosadašnjih ograničenja u štampanju iz *Windows* aplikacija. *TrueType* u Beta verziji paketa nije kompletno realizovan. Izbor fontova je manji od obećanog (trebalo bi da ih je 13, prema izjavama predstavnika „Microsoft"-a), a ime i greška u nekim od njih. Naravno, ni ovde nema jugoslovenske latinice (a ako ćemo pravo, nema ni ćirilice).

File Manager je uvek bio slaba tačka paketa. Još od pojave prve verzije *Windows*-a i DOS Executive programa, rad sa datotekama nije bio dovoljno dobro rešen. Čak su i DOS programi tipa *Norton Commander* ili *PC Tools* davali veći komfor, a svakako su bili korisniji, ali ni očigledni nedostaci nisu smetali da se u *Windows* u *3.0* **File Manager** već dosta ozbiljno koristi. Korisnici su otkrili da mogu paket aktivirati tako da se ne budi u **Program Manager**-u, već u **File Manager**-u, i time dobili jednu vrstu DOS školjke. Istina, mogućnosti rada sa datotekama bile su vrlo ograničene, ali nije se moralo izlaziti u DOS da bi se obavila neka manja operacija.

U verziji *3.1* **File Manager** je bitno poboljšan. Najznačajnija je mogućnost da se vidi više od jednog direktorijuma, stablo, ikone u raznim kombinacijama. Lakši je pristup mrežnim diskovima, i podržan je istovremeni pristup datotekama. Veći je i komfor u radu, jer je **Manager** pregledniji, prikazuje više podataka i možete menjati font kojim se ispisuju nazivi datoteka.

OLE (Object Linking and Embedding) je u osnovi mehanizam za uključivanje više tipova podataka u jedan dokument. U *Windows*-u *3.0* podatke iz druge aplikacije unosili ste u dokument preko Clipboard-a. **Edit/Cut** ili **/Copy** prenosili bi ih na Clipboard, uz nužne transformacije formata, a zatim bi komandom Paste bili uključeni u dokument koji se trenutno obrađuje. Na taj način ne može se ostvariti veza dva dokumenta, pa promena jednog od njih ponovo morate sami prenositi. Sa OLE-om je drugačije: izmene izvornog dokumenta automatski se ogledaju i u ciljnom dokumentu! Ako, na primer, kliknete mišem na uvezenim podacima, naći ćete se u aplikaciji za koju je vezan drugi dokument. OLE u Beta-verziji *Windows*-a postoji samo u *Paintbrush* i *Cardfile* programima. Postojeće aplikacije ne mogu koristiti njegove sposobnosti, ali će u novim izdanjima.si gurno postojati.

Sigurno ste mnogo puta videli poruku „Unrecoverable Application Error". UAE se pojavljuje kada aplikacija pokuša da prenese neočekivani parametar ili se ne pridržava pravila za korišćenje memorije u multitasking okolini. *Windows 3.1* značajno je doteran, pa se ove greške mnogo ređe javljaju.

U beta verziju uključen je program *Dr Watson* koji odje nad događanjima u programu. Njegova dužnost je da u trenutku kada se pojavi UAE snimi stanje paketa, memorije i procesora (jedna vrsta *Post Mortem Analysis*) i izveštaj smesti u datoteku DRWATSON.LOG. „Microsoft" je zahvalan za izveštaje koje mu dostavljaju testeri programa, pa možete okušati svoju sreću (naravno, ako uspete da nabavite *Windows 3.1*). Jedine UAE greške koje su se javljale u toku našeg dosadašnjeg rada bile su kod nekih *shareware* aplikacija.

WINDOWS 3.1 NOVITETI

Tehnička poboljšanja

- Veća brzina rada
- FastDisk drajver za disk, zaobilazi BIOS i DOS
- Manje katastrofalnih grešaka (UAE)
- *TrueType* sistem za fontove
- Povezivanje dokumenata (OLE)
- Podrška za rad na LCD i Gas-Plasma ekranima
- Poboljšani drajveri za laserske štampače
- Kod programa može se smeštiti u ROM memorije
- Podrška za više od 16 MB memorije

Program Manager

- Veći izbor boja
- Screen Saver, za čuvanje ekrana u vreme neaktivnosti
- Veća kontrola nad serijskim portovima
- Lakše instaliranje drajvera za štampače

File Manager

- Može se videti više od jednog direktorijuma u jednom trenutku
- Više informacija o sadržaju direktorijuma
- Bolji rad sa mrežnim diskovima
- Bolje rešeno formatiranje diskova

Rad na mreži

- Pouzdane mrežne veze
- Podrška za istovremeni pristup datotekama
- Moguć rad administratora mreže
- Podrška za OS/2 način imenovanja datoteka
- Podrška za rad sa read-only datotekama

ZAOBIĐENI DOS

"Microsoft"-ovi predstavnici su za *Windows 3.1* obećali novu tehnologiju rada sa tvrdim diskom, *FastDisk*. To je drajver koji se direktno obraća kontroleru diska, zaobilazeći DOS i BIOS servise. Može da radi u Protected modu 386 procesora, i to tako što presreće sve pozive INT 13H za BIOS disk servise. Ova tehnologija bi trebalo da obezbedi brži pristup disku, što je važno zbog stalnog prebacivanja delova memorije na disk.

Moramo priznati da nismo videli ništa u radu *Windows*-a što bi se ekskluzivno moglo pripisati ovom unapređenju. Povećanje brzine je rezultat promena i u jezgru programa, i grafičkog interfejsa, i modula za pristup disku zajedno. Program je testiran na računaru koji ima SCSI disk i inteligentan kontroler. Instaliran je bio i poseban drajver koji je već sam obavljao presretanje INT 13h. U svakom slučaju, sve je radilo bez problema, nije smetalo jedno drugom, što je svakako važno.

Windows 3.1 pokazuje značajan napredak u odnosu na prethodnu verziju. Neka poboljšanja mogla su se očekivati, a neka predstavljaju iznenađenja. Ne može se međutim izreći konačna ocena pre nego što se vidi konačna verzija proizvoda, delovi paketa očigledno nisu dovršeni, niti dotereni do njihovog konačnog izgleda. Još uvek ostaje mesta za kritiku „Microsoft"-a, jer su ostale polužavršene stvari kao što je File Manager ili podrška nacionalnim setovima znakova. Još uvek ostaje dosta prostora za nezavisne proizvođače dodataka, kao što su „Symantec", „Becker" ili „hDc".

Moramo priznati da se nismo ni nadali revoluciji ravnoj onoj pojavi *Windows 3.0*. Evolucija je međutim bila u dobrom pravcu i *Windows 3.1* biće, po svemu sudeći, još privlačniji i korisniji. Pojava novih i poboljšanja starih aplikacija učiniće da *Windows* postane obavezan deo gotovo svakog PC sistema.

POMOĆ PRI INSTALACIJI

Windows programi tražiće sva više i shodno tome biće sve veći. Stoga nije ni čudo što se isporučuju na pola tuceta ili više disketa. Da bi smanjio broj disketa, isporučilac programa ide sve više na to da programe komprimuje. Tokom instalacije programi se „raspakuju" i smeštaju na hard disk.

Sa novim SDK-om sve to ide nešto jednostavnije, jer on sadrži dva DOS programa pod imenima COMPRESS.EXE i EXPAND.EXE. Program COMPRESS pakuje datoteke po Lempel-Ziv algoritmu, dok ih EXPAND ponovo vraća u prvobitni oblik.

Ko pravi instalacioni program za *Windows* može se poslužiti COMPRESS-om da bi ustedeo diskete. Za vreme instalacije na hard disk biće mu od pomoći biblioteka LZEXPAND.DLL. U ovom DLL-u krije se 10 funkcija sa algoritmima za raspakivanje po Lempel-Zivu. Za raspakivanje jedne jedine datoteke može instalacioni program na primer da komprimovanu datoteku i ciljnu datoteku otvori sa **LZOpenFile**. Pozivom **LZCopy** kopirate datoteku, koja će osim toga biti i raspakovana. Zatim instalacioni program zatvara obe datoteke sa **LZClose**. LZEXPAND sadrži još i funkcije za rad sa grupama datoteka i za rad sa datotekama koje se sastoje od kombinacija raspakovanih i neraspakovanih segmenata.

3.1 ILI NE, PITANJE JE SAD

Windows 3.0 imao je jedan teko snažan uticaj na PC svet da su mnoge softverske kuće svoje programe prilagodile i za *Windows* okruženje, a počelo je i pisanje izvornih *Windows* programa. Ali, posle predstavljanja *Windows*-a *3.1* verovatno će mnogi korisnici ostati na verziji *3.0*. Zato će se softverske kuće verovatno naći u dilemi da li da uvedena proširenja primene ili ne. Na sreću, odluku nije teško doneti. Nove mogućnosti *Windows*-a *3.1* možete ipak koristiti i vaše programe izvršavati i u *Windows*-u *3.0*. Za ovo je delimično zaslužno dinamičko povezivanje. Dodajte vašem programu OLE, DDEML i prozor sa dijalogom za utvrđivanje verzije *Windows*-a i isporučite ga sa odgovarajućim DLL-ovima.

Najvažnija stvar je testi da li vaš program može da koristi *TrueType* fontove. Za to će verovatno biti potrebne male izmene koda, a možda ni toliko. I na kraju ne zaboravite – nađite dovoljno slobodnog vremena. Dok vi budete izgustirali *Windows 3.1*, tada već ni *Windows NT* (*new technology*) neće biti daleko!

Microsoft System Journal (Nov/Dec 1991)

Priredio Kragl Aleksandar

DODACI ZA LOTUS

Dobar deo snage Lotusa 1-2-3 leži u dodacima za najrazličitije namene koje proizvode nezavisne kompanije. Takvih programa ima na stotine, a mi smo za vas izabrali devet za koje smatramo da vam mogu olakšati svakodnevni život sa Lotusom.

Vladimir Stamenović

Već sam nekoliko puta u svojim člancima spominjao dodatne programe koji rade u okviru Lotusa 1-2-3. Ovog puta reč je o proizvodima nezavisnih proizvođača softvera koji su našli svoje mesto na tržištu. Trenutno postoji na stotine takvih programa za najrazličitije namene. Možda bi se mogli grubo podeliti u nekoliko grupa.

TRI GLAVNE GRUPE

Obzirom da na severnoameričko i britansko tržište odlazi najveći deo proizvodnje *Add-in* programa, a i na specifičnost njihovih poreskih, platnih i finansijskih sistema, preskočićemo možda najveću grupu dodatnih programa za 1-2-3. Reč je o programima koji korisnicima omogućuju lako praćenje svojih poreskih i sličnih obaveza, a ogroman broj programa bavi se finansijskim i komercijalnim poslovanjem, kako malih tako i onih najvećih firmi. Čini mi se da sam već negde napisao da velika većina američkih kompanija svoje poslove obavlja pomoću *Lotusa 1-2-3* i odgovarajućih dodatnih programa. U Beogradu je to slučaj sa, na primer, McDonald-som i Hayatt-om.

Postoji i velika grupa dodatnih programa koji se bave problemima linearnog programiranja, optimizacije, te pomoći u rešavanju raznih problema. Iz ove grupe na testiranje sam dobio jedan od najboljih programa, *What's Best! Personal Version 1.6*, *What If Analyst Premium Edition* i *What If Solver*.

Treća i meni najzanimljivija grupe *Add-in* programa su pomoćni programi. Oni omogućuju da se unapredi rad sa printerima, ekranima, *expanded* memorijom, makroima (dibageri) itd. Na testiranju ih je bilo nekoliko: *Beyond 640, Guardian (Extracopy), Rescue Plus, Macro Editor/Debugger, Sideways* i *321 Gosub*.

Dakle, o ovih devet dodatnih programa za *Lotus 1-2-3* će biti reči u daljem tekstu. Zajedničko im je da su predviđeni za rad sa 1-2-3 verzije 2.0, 2.01, 2.2, a neki rade i sa 1-2-3 verzija 2.3 i 3.x, kao i sa svim verzijama *Symphony*. Mislim da kod nas bar 80% korisnika 1-2-3 radi na verziji 2.01 (i to često kopiji, na žalost), pa smatram da će ovaj moj izbor biti na zadovoljstvo većine korisnika 1-2-3. Trudiću se da, tamo gde treba, naglasim neke eventualne posebnosti koje se odnose na 1-2-3 verzija 3.x i *Symphony*.

POKRETANJE ADD-IN PROGARMA

Dodatni programi, na aktiviraju, pridružuju glavnom programu kroz takozvani *Add-in Manager*. U Verzijama 2.2, 2.3, 3.x, kao i u *Symphony*, on je uključen u glavni program i poziva se /**Add-in** komandom iz glavnog menija, odnosno kod *Symphony* kroz SEVICES (F9) meni komandom **Application**. Kod ranijih verzija 1-2-3 2.0 i 2.01 toga nema. Zato je u svim dodatnim programima predviđena mogućnost rada i sa tim verzijama. Naime, prilikom instalacije na disk prebaciće se sa originalne diskete i *Add-in Manager*. Njega ćete pozivati kombinacijom tastera Alt-F10, koja inače radi i sa verzijama 1-2-3 2.2 i 2.3. Treba napomenuti da aktiviranjem *Add-in Manager*-a u 1-2-3 verzijama 2.0 i 2.01 trošimo oko 20 k RAM-a.

Dakle, istovremenim pritiskom na tastere Alt i F10 pozivamo *Add-in Manager* meni. Prva komanda je **Attach**. Kad je aktiviramo, u prvom redu iznad tabele dobićemo spisak svih raspoloživih dodatnih programa (datoteke .ADN). Na nama je da odaberemo odgovarajuće ime programa i pritisnemo Enter. Možemo program pridružiti nekom od funkcijskih tastera. U svakom slučaju, komandom **Attach** smo učitali dodatni program sa diska u memoriju računara i potpuno ga integrisali sa 1-2-3. Brišemo ga iz memorije komandom **Detach**. Ako imamo više *Add-in* programa u memoriji, možemo ih aktivirati jedan po jedan komandom *Invoke ime programa*. Sve dodatne programe možemo obrisati iz memorije komandom **Clear**. Postoji mogućnost da se jedan ili više *Add-in* programa učitavaju automatski pri startovanju *Lotusa 1-2-3*. To omogućuju opcije komande **Setup** iz *Add-in Manager* menija. Na kraju menija je komanda **Quit** koja služi za izlaz iz *Add-in Manager*-a.

Dodatni programi uglavnom imaju sopstvene menije koji se aktiviraju po pozivu programa. Oni su potpuno u Lotus stilu – rade identično kao i originalni meni 1-2-3. Kroz njih ćete obavljati sve poslove koje omogućuje i nudi dodatni program.

No, krenimo redom da vidimo šta nude *Add-in* programi koje za sada imamo na raspolaganju.

BEYOND 640 Version 1.02

Proizvođač: Intex Solutions Inc.
161 Highland Ave.
Needham, MA 02194, USA
Tel: 617-449-6222
Cena: ispod 100 $
Memorija: zauzima 4 K RAM-a, podržava LIM 3.2 i 4.0 *expanded* memoriju
Okruženje: *Lotus 1-2-3* verzija 2.x i DOS 2.0 ili veći



Ovaj program je iz kategorije onih koji pomažu u iskorišćenju raspoložive *expanded* memorije u računaru. Spada u red najprodavanijih programa ovog tipa.

Beyond 640 se isporučuje na jednoj disketi 5.25" ili, po zahtevu, na jednoj 3.25" disketi. Uz nju ide i kratko uputstvo za rad na svega 13 stranica. No, na to malo prostora rečeno je sve što treba da se kaže o programu, pa korisnici verovatno neće imati problema sa upotrebom.

Instalacija se sastoji u kopiranju diskete u 1-2-3 poddirektorijum. Ako nemate *Add-in Manager* u vašoj kopiji 1-2-3, zadaćete komandu ADD MGR 123.SET. Time ćete dopuniti postojeći drajver set i omogućiti sebi korišćenje *Add-in Manager*-a. Instalacija je moguća i na disketu za one koji koriste 1-2-3 sa diskete. Da biste napravili mesta, moraćete izbrisati 123.HLP datoteku.

Pokretanje *Beyond 640* ide preko *Add-in Manager* menija. **Proizvođač** strogo preporučuje da se program automatski učita pri startovanju 1-2-3.

U čemu je štos kod ovog dodatnog programa? Korisnicima *Lotusa 1-2-3* verzije 2.x se verovatno događalo da i pored 640 K RAM-a i dosta raspoložive *expanded* memorije dobiju poruku od 1-2-3 „Out of memory" ili „Memory full" pri kreiranju većih radnih tabela. Taj problem se pojavljuje zato što 1-2-3 mora da koristi konvencionalnu memoriju za čuvanje delova tabele. Ako koristite i neki rezidentni program koji zauzima svoj deo RAM-a, situacija može biti još gora. *Beyond 640* pomaže u rešavanju ovakvih problema tako što već deo ili skoro celu tabelu smesta u *expanded* memoriju. Tako ostaje više memorije za veće radne tabele i aplikacije.

Pri upotrebi *Beyond* jedino što treba odrediti je minimalna količina raspoloživog RAM-a od 0 do 640 K, nakon čega će sve biti prebacivano u *expanded* memoriju. Moguće je upotrebiti iskoristiti do 4 MB *expanded* memorije. Dakle, kada tabela koju kreira korisnik bude toliko da se ostatak slobodnog RAM-a približi minimumu određenom u *Beyond* program automatski počinje da prebacuje sadržaj tabele u *expanded* memoriju. Tako stalno čuva određeni minimum RAM-a. Koliki će taj minimum biti zavisi od potreba korisnika. Ako radite, pored 1-2-3, sa rezidentnim programima, kao što su *Allways*, *Wysiwyg*, *SideKick*, morate i njih uzeti u obzir. Na primer, *Wysiwyg* dodatni program za 1-2-3 verzija 2.3 zauzima oko 100 K RAM-a. Korisnik može u *Beyond 640* programu da zahteva minimalni raspoloživi RAM od 100 K, nakon čega će podaci iz tabele ići u *expanded* memoriju.

Neko će možda odlučiti da minimum bude 640 K. U tom slučaju na raspolaganju je ceo RAM za 1-2-3, DOS, rezidentne ili dodatne programe. Cela tabela onda ide u *expanded* memoriju. Zvuči razumno i prihvatljivo. Međutim, time se dosta usporava rad, jer je *expanded* memorija znatno sporija od RAM memorije. No i tu ima razlike. Ako je *expanded* memorija na dodatnoj kartici, rad je sporiji nego

kada je ona direktno na matičnoj ploči. Sa dru ge strane, dobijamo na prostoru, pa možemo da pravimo veće i komplikovanije tabele.

Mislim da bi se uz malo eksperimentisanja na konkretnom slučaju lako došlo do optimalnih rešenja.

Svima koji imaju problema pri pravljenju većih tabela tipa "memory full" i slično, a raspolažu *expanded* memorijom, preporučujem *Beyond 640*, kao jedno od zgodnih rešenja.

RESCUE PLUS

Proizvođač: Intex Solutions Inc.
161 Highland Ave
Needham, MA 02194, USA
Tel: 617 449 6222
Cena: 130 $, a 150 $ za program koji podržava 1-2-3 ver. 3.0
Memorija: zauzima 256 K RAM-a
Okruženje: *Lotus 1-2-3* verzija 2.x, sve verzije *Symphony* i DOS 2.0 ili veći

Rescue Plus spada u "file and disk recovery" programe. On je u stanju da obnovi oštećene i obrisane datoteke, a namenjen je specijalno za *Lotus 1-2-3* i *Symphony*. Prema kritikama i testovima koje sam video, najbolji je među svojim konkurentima, pa čak i od *Norton Utilities File Fix* programa, naravno samo za 1-2-3 i *Symphony* datoteke.

Rescue Plus se isporučuje na jednoj disketi 5.25" ili, po zahtevu, na jednoj 3.25" disketi. Uz nju ide i kratko uputstvo za rad, *User Manual*, na 42 strane. Nije baš najdetaljnije, ali uz malo vežbe korisnici verovatno neće imati problema sa upotrebom.

Program *Install* sa diskete će iskopirati sve datoteke *Rescue Plus* programa u 1-2-3 poddirektorijum. VRLO JE VAŽNO da ne instalirate *Rescue Plus* na disk nakon što ste izgubili tabelu. Ako to učinite, može se desiti da datoteke *Rescue Plus* programa budu zapisane preko oštećenih podataka, a onda nema popravke. Ako je takva situacija, startujte program direktno sa diskete.

Rescue Plus postoji u dve paralelne verzije. Jedna, ova koju ću vam predstaviti, namenjena je za popravke izgubljenih ili oštećenih datoteka sa diska u *Lotusu 1-2-3* verzija 2.x i *Symphony*. Druga verzija se zove *Rescue Plus edition II* i namenjena je istom poslu za 1-2-3 verzije 3.x datoteke.

Na disketi ima ukupno pet programa koji čine *Rescue Plus*. Koji od njih ćete upotrebiti zavisi od načina oštećenja datoteke. Ako greške u datoteci nisu posledica kvara na disku ili njegovog FAT-a, onda će program *Rescue* biti najbolje i najbrže rešenje problema: napraviće oštećeni fajl čitljivim. Interesantan je metod utvrđivanja tipa greške koji preporučuje proizvođač. Naime, ako se oštećena datoteka vidi u katalogu po izdavanju DOS komande Dir, ako njena veličina izgleda normalna i ako ona veličina datoteke ostane ista pri kopiranju na drugi disk ili disketu, onda je velika verovatnoća da oštećenje nije posledica kvara na disku ili FAT-u. U tom slučaju, kao što rekoh, *Rescue* završava posao.

U drugim slučajevima, kada je datoteka rasuta po disku ili kada je treba rekonstruisati iz njenih delova, posao će završiti programi *ResQDisk*, *ResQView* i *ResQFile*. Tu je i program *ResQPass* koji odlično pomaže ako zaboravite pasvord zaštitu za tabelu.

Rescue je možda jedini program koji je u stanju da popravi sve 1-2-3 i *Symphony* datoteke. One mogu biti oštećene na razne načine. Bitno je da oštećenja nisu posledica kvara na disku i FAT-u, a *Rescue* će bajt po bajt pretražiti disk i naći sve delove datoteke, proveriće da li su to delovi Lotusovih datoteka i onda ih spojiti u novu, ispravnu. Program obavlja posao dosta brzo. Možete i štampati izveštaj o popravljenom fajlu u kome se tačno vidi gde je bilo oštećenje. Korisnici koji rade u tabelama zaštićenim lozinkom mogu da odahnu: *Rescue* uspešno popravlja i takve datoteke.

Programi *ResQDisk*, *ResQView* i *ResQFile* zajedno pomažu da se rekonstruiše oštećena datoteka. Čudno je da nisu integrisani u jednu celinu, ali važno je da rade posao. Kod popravke oštećenih datoteka koje *Rescue* ne može da vrati, postupak je sledeći: Program *ResQDisk* pretražuje ceo disk u potrazi za svim klasterima na disku koji sadrže podatke iz tabela, čita veličinu i početne klastere postojećih *Lotusovih* datoteka na disku. Nakon toga sastavlja izveštaj o radu i štampa ga na štampaču. Posle *ResQDisk*-a, programom *ResQView* možete pregledati sve klastere da se uverite da li u njima ima nešto od izgubljenih podataka. Na kraju, *ResQFile* program će od svih klastera koji sadrže podatke iz oštećene ili izgubljene datoteke sklopiti novu, ispravnu, čitljivu datoteku.

Moram priznati da mi nije odmah pošlo za rukom da rekonstruišem jednu datoteku koju sam namerno oštetio. No, nakon nekoliko pokušaja, ukapirao sam sistem rada i posle nije bilo problema. Uglavnom sam na mojim primerima uspevao da povratim datoteku. Smetalo mi je jedino što se ceo posao obavlja ručno.

Nisam pravio poređenje sa *Norton Utilities File Fix* programom koji radi iste stvari, ali neki američki izvori tvrde da je za popravku oštećenih i izgubljenih Lotusovih datoteka najbolje rešenje *Rescue Plus*.

Dakle, za sve one koji dosta i uglavnom rade sa *Lotusom 1-2-3* ili *Symphony*, preporučujem *Rescue Plus* kao efikasno obezbeđenje od gubljenja podataka.

GUARDIAN (EXTRACOPY) 2.0

Proizvođač: Intex Solutions Inc.
161 Highland Ave
Needham, MA 02194, USA
Tel: 617 449 6222
Cena: ispod 100 $
Memorija: zauzima 28 K RAM-a
Okruženje: *Lotus 1-2-3* verzija 2.x, sve verzije *Symphony* i DOS 2.0 ili veći

Iz iskustva znam koliko je lako izgubiti tabelu na kojoj se satima radi. Razloga ima dosta. Najčešće, nakon rada korisnik izađe iz 1-2-3 ili *Symphony*, a da nije snimio tabelu na disk, ili učita novu tabelu sa diska, a da nije snimio onu na kojoj je radio. Osim toga, događa se meni nekoliko puta da u sekundi nestane električne energije. Sasvim dovoljno da se svi podaci obrišu iz memorije. Naravno, ima još mnogo načina da se izgubi tabela uz koju je neko proveo dosta vremena kreirajući je. Evo, upravo dok sam pisao ovaj tekst, moj dvadesetomesečni sin je, igrajući se, pritisnuo interesantno zeleno dugme na mom laptopu i ugasio ga. Izgubio sam deo teksta, ali na sreću je *Guardian* radio svoj posao dok sam ja pisao (*Symphony* tekst procesor).

Ovaj program, koji se do nedavno zvao *Guardian*, a sada mu je naziv *Extracopy*, predstavlja odlično osiguranje od gubitka datoteka kreiranih u 1-2-3 ili *Symphony*. On radi u pozadini tabele, automatski pravi njen bekap dok se korisnik koncentriše na rad, oslobođen brige da li je snimio svoju tabelu na disk.

Program stiže na jednoj 5.25" disketi ili na 3.5", ako se tako poruči. Uputstvo za upotrebu je malo, na 29 strana, ali sasvim dovoljno da bi se *Guardian* efikasno koristio. Može se instalirati i na disk, u poddirektorijum 1-2-3, i na disketu sa 1-2-3. U drugom slučaju, da biste napravili mesta na disketi, morate obrisati 123.HLP datoteku. Instalacija se sastoji od prostog kopiranja svih datoteka sa *Guardian* diskete na disk.

Učitavanjem ovog dodatnog programa dobijamo na ekranu *Guardijan*ov meni, takozvani **Guardian Settings Screen**.

Kroz meni ćemo izvršiti sva podešavanja prema našim potrebama, a na ekranu ćemo videti stanje svih raspoloživih opcija u programu.

Prva i najinteresantnija komanda je **Auto**. Kroz nju možete odrediti način na koji će *Guardian* automatski snimati tabelu na disk. To može biti neki vremenski interval od 1-120 minuta, nakon koga će tabela biti snimljena. Snimanje na disk se obavlja u pozadini, a samo na kratko vidite poruku o tome u vrhu ekrana. Za početak možete odrediti kraći interval, na primer 10 minuta, a kasnije ga prilagodite svom načinu rada.

Snimanje na disk može biti određeno brojem pritisnutih tastera na tastaturi (25-10000) ili kombinacijom ova dva načina. Ako, na primer, odredite da se tabela snima na disk nakon 150 pritisnutih tastera, to će se stvarno i dogoditi. Guardijan broji sve tastere koje pritisnete. Kombinacija vremenskog intervala i broja aktiviranih tastera je najsigurnija. Kad je ispunjen jedan od ova dva uslova, tabela će biti snimljena na disk.

Kroz **Auto** komandu može se odrediti poddirektorijum gde će datoteke biti snimane, kao i pod kojim imenom i ekstenzijom će to biti urađeno.

Važno je napomenuti da *Guardian* snima na disk samo kada je 1-2-3 u READY modu, a *Symphony* u SHEET, DOC ili FORM modu. To znači da za vreme izvršavanja neke makro procedure, prikazivanja grafika ili rada u meniju, *Guardian* neće snimati tabelu na disk, već će čekati READY mod. Datoteka koju je *Guardian* snimio neće biti obrisana sve dok se ne izda komanda /File Save iz 1-2-3 menija, odnosno SERVICES File Save iz *Symphony* menija.

Veoma je zanimljiva opcija **Version**. Kroz nju možete pratiti razvoj vaše tabele – aplikacije i to u maksimalno 50 verzija. Naravno, broj verzija možete sami odrediti. Verzije će imati isto ime, ali će se ekstenzija menjati: V01, V02, V03 i tako dalje do maksimalnog zadatog broja verzija. Nakon toga briše se najstarija verzija, a sve ostale se prenumerišu da se oslobodi mesto za novu verziju.

Sve ove datoteke zauzimaju dosta mesta na disku, pa je uz *Guardian* isporučen i program za kompresiju datoteka. Zove se LHARC. Dato je i uputstvo za njegovu upotrebu u datoteci LHARC.MAN.

Preporučujem da različite verzije, kompresiju datoteka na disku radite u različitim poddirektorijumima, da bi se izbegle eventualne greške.

Na kraju da kažem da mi se *Guardian* neobično dopada. Zauzima malo RAM-a, veoma je jednostavan, a sigurno obezbeđuje korisnika od gubitka tabele. Osim toga, za one koji razvijaju razne aplikacije, mogućnost praćenja razvoja aplikacije kroz pravljenje više verzija izuzetno je korisna. Preporučujem ga svim korisnicima 1-2-3 i *Symphony*.

MACRO EDITOR/DEBUGGER 1.05

Proizvođač Personics Corp
63 Great Rd
Maynard MA 01754 USA
Tel 800 445 3311 508 897 1575
Cena ispod 200 $
Memorija zauzima 62 K RAM-a
Okruženje *Lotus 1-2-3* verzija 2.x (uključujući 2.3) i DOS 2.0 ili veći

Nije mi običaj, ali moram odmah da kažem da je ovaj program pun pogodak i pravi spas za programere u 1-2-3. Slično „pravim" programskim jezicima (paskal, C), najzad smo i mi korisnici, kako neki tvrde (PC Magazine, Craig Stinson) „najraširenijeg jezika za razvoj aplikacija na svetu", dobili pravi editor i dibager.

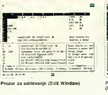
Zahvaljujući Lotusovoj koncepciji razvoja koja je omogućila ogromnom broju nezavisnih proizvođača softvera da proizvodi dodatne programe za uglavnom 1-2-3, dobili smo jedan izuzetan program koji će znatno pomoći u radu i efikasnosti pri pravljenju aplikacija. Na taj način je izbegnuto komplikovanje osnovnog programa, koje srećemo u konkurentskim proizvodima. Naime, i Borlandov *QuattroPro* i Microsoft-ov *Excel* imaju neku vrstu makro dibagera, ali slobodno mogu reći, po mogućnostima ni blizu ovome što nudi Personics Corp. kroz svoj *Macro Editor/Debugger* (u daljem tekstu *MED*) *Add-in* za 1-2-3.



MED se isporučuje u kutiji u kojoj se nalaze diskete sa programom (po jedna od 5.25" i 3.5"), uputstvo za upotrebu i registraciona kartica. Uputstvo na 190 strana je dosta dobro i opisuje kompletan način rada sa ovim programom. Posebno je koristan *Tutorial* deo u kome je prilično detaljno opisano sve što treba da bi korisnik prihvatio način rada sa programom. Na zadnje dve strane knjiga je dat *Quick Reference*: pregled svih komandi iz menija, kao i tastera koji se koriste u radu sa *MED*-om. Zanimljivo je i da su opisane i neke nedokumentovane karakteristike 1-2-3.

Za instalaciju programa na disk u 1-2-3 po direktorijum služi program *Install* koji prebacuje sve datoteke na disk. Postoji i program *REM_MED* za deinstalaciju programa sa diska.

MED se pokreće iz *Add-in Manager*-a, kao i svi dodatni programi. Može se pridružiti nekom od funkcijskih tastera i pozivati po potrebi.



Prozor za editovanje (Edit Window)

Drugi način je da, nakon učitavanja, pritisnete tastere Shift-F2 (Step mod u 1-2-3) i tako aktivirate *MED*. Odmah nakon startovanja vidi se karakterističan *MED*-ov dibager prozor na donjem delu ekrana, preko tabele. Uvek možete pritiskom na taster / (kosa crta, engl. slash) pozvati *MED*-ov meni.

MED ima tri prozora. Jedan je za dibager, drugi za editovanje makroa, a treći **Breakpoint** za prekidne tačke.

Prozor za dibager je podeljen na tri dela: prostor za makro procedure, za poruke i statusni deo. U prostoru za makroe vidi se ona procedura koja treba da se izvrši. Na vrhu je ona makro ćelija u kojoj je instrukcija koja se trenutno izvršava. Obeležena je pointerom. Osim toga, ako ste označili tačke prekida procedura (*breakpoints*), one će biti označene u makro kodu. Ako dibager naiđe na grananje, na primer {branch} komandu, odmah ispod će naznačiti gde to grananje, odnosno prenos kontrole makroa, ide.

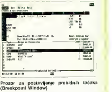
Prostor za poruke u dibager prozoru je na dnu ekrana i u njemu se prate informacije o trenutnom statusu *MED*-a, poruke upozorenja, greške i prekida rada.

Statusni prostor sadrži informacije o tome koja makro procedura se izvršava, njeno ime (naziv oblasti ili adresa ćelije), adresu pozivnog makroa (kad se izvršava potprogram), statusni indikatori i aktivne tačke posmatranja (engl. *Watchpoints*).

Iz dibager prozora, pritiskom na F2 taster, možemo preći u edit prozor. U njemu je moguće kreirati nove makroe ili prepravljati postojeće. I ovaj prozor je podeljen na tri dela: prozor za oznake, nazive za makro procedure i za komentare koda makroa. Sve je ovo napravljeno dosta zgodno. Na primer, kada pišete novu makro proceduru, *MED* automatski imenu dodaje prefiks labele, definiše ime oblasti i pridružuje ga makrou u susednoj ćeliji. Ovo bi inače u 1-2-3 radili komandama **/Range Name Create Labels Right**. Ako obrišete postojeći naziv makroa u prozoru za nazive, *MED* će obrisati odgovarajuće ime oblasti (**/Range Name Erase**), a ako ga promenite, *MED* će obrisati staro ime oblasti i napraviti novo (**/Range Name Erase** pa onda **/Range name Create**). Ušteda u vremenu i komfor u radu su očigledni.

U prozoru za editovanje makroa se vidi kolona iz labele sa makro procedurom. Možete ispravljati i prepravljati po želji ne samo makro instrukcije već i string i numeričke formule. Isto tako, možete pisati potpuno nov kod. Desno od ovog prozora je prozor za komentare. Naime, pri pisanju programa i makro procedura izuzetno je korisno napisati sa strane i komentar. Tako ćete kasnije lakše znati šta ste radili, a eventualne ispravke uradićete mnogo brže.

Treći prozor koji postoji u *MED*-u je onaj koji sadrži tačke prekida izvršavanja makro



Prozor za postavljanje prekidnih tačaka (Breakpoint Window)

procedura (engl. *Breakpoint Window*). Zadavanje tačaka prekida programa je izuzetno korisna mogućnost. To vam omogućuje da pokrenete program koji će se prekinuti kada je zadovoljen neki unapred određeni kriterijum ili događaj.

Dakle, makro koji ste pokrenuli se zaustavlja na mestu gde ste zadali tačku prekida. *MED* izdaje poruku o tome, a ako pritisnete ENTER, biće označena ona tačka prekida, iz kataloga svih tačaka, koja je dovela do prekida izvršavanja makro procedure. Inače, može se postaviti maksimalno 9 tačaka prekida, koje su razvrstane u tri tipa. Izvršna tačka prekida stopira makro kada dođe do određene ćelije. Prekid se može zadati i kada sadržaj neke određene ćelije promeni vrednost. Vrlo je zanimljiva i korisna mogućnost da se izvršenje programa prekine kada se ispuni neki zadati uslov.

U *MED*-u možete započeti izvršavanje makroprocedure u bilo kojoj tački. Samo namestite pointer u odgovarajuću liniju koda i startujte na jedan od tri raspoloživa načina: instrukciju po instrukciju, nekoliko njih ili rad normalnom brzinom do prve tačke prekida. U svakom momentu, kada je došlo do zastoja u izvršenju makroa, možete preći u EDIT prozor i ispraviti ili dopuniti kod, a onda se vratiti u dibager i nastaviti kontrolu.

Sve u svemu, *MED* je izgleda predvideo sve što treba jednom programeru u 1-2-3. S njim je kontrole napravljenih makro procedura zaista efikasna i korisnicima će uštedeti sate i sate rada. Siguran sam da će svaki programer u 1-2-3 biti oduševljen mogućnostima ovog *Add-in* programa.

Dakle, ako ste ozbiljan korisnik i proizvođač makro procedura, Personics-ov *Macro Editor/Debugger* vam je neophodan.

Sideways

Proizvođač Funk Software Inc
222 Third St
Cambridge MA 02142 USA
Tel 617 497 6339
Cena oko 90 $
Memorija 80 K RAM-a
Okruženje *Lotus 1-2-3* verzije 2.x, 3.x, *Symphony* i DOS 2.0 ili veći

Sideways je na tržištu odavno, iskoristiću priliku da ga spomenem, mada nemam najnoviju verziju već staru 3.0. Nastao je kao odgovor na potrebe korisnika 1-2-3 da štampaju široke labele sa mnogo kolona. Takva labela na papiru obično se naziva čaršav. Sistem je jednostavan: *Sideways* okrene **print range** za 90 stepeni i štampa popreko papira. Pošto se takve labele štampaju na beskonačnom kompjuterskom papiru, širina labele, odnosno nje

na dužina na papiru je ograničena samo veličinom tabele.

Program je samostalan kada se štampaju datoteke 1-2-3, a dodatni u *Symphony*. Dakle, pokrenete *Sideways* sa sw123.exe datotekom i dobićete *Sideways* meni koji je potpuno u Lotus stilu, a poziva se sa / (kosa crta). Na ekranu se vidi **Print Settings**, odnosno sva podešavanja koja su urađena ili se mogu uraditi. Iz *Symphony* ceo isti posao uradili kroz komande **[F9] Aplication Attach (Invoke) Sideways.app**.

No, pre startovanja treba da odredite printer sa kojim raspolažete kroz instalacioni program SWSETUP. Nakon starta komandom / **Interface** odredite port na koji je priključen štampač. Dimenzija papira su obično standardnih 8×11 inča za beskonačni papir sa perforacijom. Kroz opcije (**Form-size**) možete odrediti bilo koje dimenzije papira. Obzirom da *Sideways* štampa popreko papira, onda se i margine određuju u odnosu na tu činjenicu. Naime, gornja margina je razmak od desne ivice papira (kada ga posmatramo normalno) do početka teksta. U margine spada i podatak o tome kako će program tretirati kraj strane (**Perf-skip**): da li će preskočiti perforaciju ili će štampati preko nje. Možete zadati i neke dodatne parametre koji će vašu tabelu učiniti lepšom kada se odštampa. Na primer, deo tabele može biti štampan masnim (bold) slovima, proširenim ili smanjenim slovima, podvučeno itd.

Pre nego počnete sa štampom, morate prvo učitati datoteku koju štampate (/**File Retrieve**). Uz ime obavezno morate navesti i ekstenziju (WKA, WK1 ili WKS). Nakon toga možete je pregledati jer tabela *Sideways*-a izgleda isto kao i originalna 1-2-3. Ne možete unositi nikakve izmene u tabelu – dozvoljeni su samo tasteri za kretanje po njoj: strelice za pomeranje pointera, PgDn, PgUp, Home i END u kombinaciji sa strelicama.

Određivanje oblasti za štampanje je isto kao u 1-2-3 – osvetljavanjem, preko adresa ili imena oblasti ćelija, a kroz komandu /**Range**.

Štampanje može biti u *draft*, *single* i *double* modu. Najkvalitetnije i najsporije je u double modu, jer tada glava štampača prelazi po dva puta preko jednog reda.

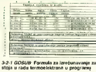
Dakle, nakon učitavanja datoteke i određivanja oblasti za štampanje, aktivirate komandu **Go** i printer počinje sa štampom. Posao možete prekinuti u svakom trenutku sa CTRL-BREAK.

Treba reći da u novim verzijama 1-2-3 2.3 i 3.x postoji mogućnost ovakvog štampanja (komanda */Print Options Advanced Layout Orientation Landscape*). Međutim, morate imati i printer koji može da štampa u *landscape* (vodoravnom) modu. Ja ga nemam i zato još uvek, kada se ukaže potreba, koristim *Sideways*.

Sideways je program bez koga se može živeti i raditi sve dok se ne pojavi potreba za štampanjem „čaršava". Onda je nezamenljiv i odlično završava posao.

3-2-1 GOSUB, Developer Edition

Proizvođač: Frontline Systems Inc.
140 University Avenue, Suite 100
Palo Alto, California 94301, USA
Tel.: 415 329-6877
Cena: oko 250 $ (Standard Edition je oko 100 $)
Memorija: zauzima 3 K (+ 50 K za kompajler) RAM-a
Okruženje: *Lotus 1-2-3* verzije 2.x, DOS 2.0 ili veći

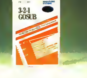


3-2-1 GOSUB: **Formula za izračunavanje zastoja u radu termoelektrana u programu**

U jednom od prethodnih brojeva „Računara" sam pisao o *Add in Toolkit*-u za 1-2-3 verzije 3.0. Sada postoji isti takav program i za verzije 2.x. Jedna od namena, mogućih primena tih alata je pisanje sopstvenih @funkcija koje se primenjuju u 1-2-3. Dodatni program *3-2-1 GOSUB* ima baš takvu namenu – proširenje biblioteke standardnih @funkcija u 1-2-3 verzije 2.x sopstvenim specijalizovanim @funkcijama. Takve funkcije će raditi mnogo brže od originalnih, a moći ćete da ih koristite u bilo kojoj tabeli. Za razliku od *Add in* Toolkit-a, nije potrebno poznavanje C-a ili asemblera, jer se funkcije i rutine prave direktno u tabeli, bez upotrebe programskih jezika!

3-2-1 GOSUB se isporučuje na po jednoj disketi oba formata, uz koje ide i uputstvo za upotrebu. Knjižica je mala, a navedene su sve mogućnosti koje nudi program. Mogla je biti i nešto opširnija, kako bi bila jasnija početnicima, da bi uz primere koji su dati na disketi mogli brže da savladaju materiju i krenu na posao. No, moram reći da *3-2-1 GOSUB* i nije namenjen potpunim početnicima. Podrazumeva se da dobro poznajete 1-2-3, pogotovu @funkcije.

Instalacija se sastoji od kopiranja sadržaja diskete u 123 poddirektorijum. Alternativa tome je da sa diskete startujete program INSTGO koji će obaviti kopiranje. Program nije zaštićen.



3-2-1 GOSUB se proizvodi u dve verzije: *Standard Edition* i *Developer Edition*. Razlika je, osim u ceni, u tome što u okviru *Developer Edition* postoji kompajler kojim se funkcija napisana u tabeli može kompajlirati i koristiti kao zaseban *Add in*. Takve funkcije omogućuju da se kompleksne kalkulacije spoje u jednostavnu naredbu. Tako, na primer, umesto mnogobrojnih funkcija koje koristite po celoj tabeli, a koje su međusobno povezane, možete napraviti jednu koju ćete pozivati svuda gde treba. Time se postiže veća kontrola funkcija i formula u tabeli, štedi vreme, jer je rekalkulacija brža, kao i memorije. Novu @funkciju ADN možete dati ili prodati drugim korisnicima koji će je koristiti baš kao i sve ostale @funkcije iz 1-2-3. Zaštićene su, jer su kompajlirane, a rade dva do tri puta brže od originalnih.

Sopstvene @funkcije se grade uz pomoć 11 funkcija – graditelja koje sadrži *3-2-1 GOSUB* i uobičajenih formula iz 1-2-3. Ključ programa je dodatna @funkcija koja se zove samo „@". Kratko ime je dato da bi i napravljena funkcija imala što kraći poziv. Kroz nju „prolazi" rutina koja je napisana u tabeli, kao i svi potrebni argumenti. Moguće je napisati @funkcije koje će rešavati razne probleme, kao što su račun sa matricama, vektorska operacije, aritmetika sa kompleksnim i imaginarnim brojevima, statističke funkcije, naučne i inžinjerske formule raznoraznih specijalnih namena, diferencijalne jednačine, optimizacija, analiza i simulacija i još što šta. Čak šta više, proizvođač tvrdi da se može napraviti @funkcija koja rešava bilo koji problem koji je moguće rešiti kroz neki programski jezik!

Pokušaću kroz jedan jednostavan primer da vam pokažem kako se može napraviti sopstvena @funkcija.

U termoelektranama „Nikola Tesla", gde radim, između ostalog vodi se evidencija zastoja u radu pojedinih blokova termoelektrane. Baza podataka je napravljena u *Lotusu 1-2-3*. Svaki zastoj u radu ima svoj početak i kraj (datum i vreme). Jedan od interesantnih podataka je vreme trajanja zastoja. Kao što se vidi na slici, formula koja taj podatak izračunava je jednostavna.

Međutim, baza ispada za svaki blok sadrži po nekoliko stotina slogova sa znatno više podataka nego što se na slici vidi. U toj situaciji ova jednostavna formula postaje teret, jer se izvršava sporije od @funkcije i zauzima više memorije. Problem se naravno može rešiti na više načina, ali je dobar primer za pravljenje sopstvene namenske @funkcije. Dakle, cilj nam je da umesto klasične formule napravimo @funkciju koja će raditi isti posao, ali brže, uz manji utrošak memorije.

Posao je jednostavan i može se urediti na više načina. Ja sam napravio dve funkcije @TRZAS i @TRZASR. Obe se sastoje samo iz dve linije. Rutine je najbolje pisati u praznoj tabeli da ne biste poremetili nešto od postojećih podataka. U prvoj liniji programa se vrši provera argumenata @funkcije. U @TRZAS je predviđeno da argumenti budu ćelije koje sadrže podatke koji ulaze u račun. Pošto ih ima 4 a sve su brojevi (datum je broj), onda u funkciji @ArgCheck treba nabrojati 4 znaka #. Druga linija objašnjava kako se vrši račun. Argumente funkcije, adrese ćelija, treba unositi tačnim redosledom, kako to zahteva formula.

U funkciji @TRZASR argument je oblast ćelija koje sadrže podatke koje ulaze u račun. Ovo je svakako povoljnija varijanta, jer je jednostavnija za unos. Naime, ne treba voditi računa o redosledu po kojem ćelije ulaze u račun. U ovoj funkciji argument je range – oblast ćelija koja se sastoji uvek od jednog reda i 4 kolone. Obzirom da svaka ćelija iz te oblasti ima svoju ulogu u formuli za izračunavanje trajanja zastoja, treba tačno odrediti redosled. Za to će odlično poslužiti @funkcija iz 1-2-3 @INDEX(oblast,kolona,red) koja vraća vrednost iz specificirane kolone i reda. Prva kolona u oblasti ima vrednost 0 (nula), sledeća 1 itd. Isto je i sa redom. Druga linija programa za @TRZASR se piše u jednu ćeliju zajedno sa trećom linijom. Na slici je ta linija razbijena na dve da bi se bolje videla. Ovaj program bi se mogao dopuniti i proverom koja bi utvrdila da li je određena oblast od jednog reda i 4 kolone, pa tek onda da se izvrši izračunavanje.

Dakle, da ponovim: najčistija je situacija kad jednu @funkciju pišete u jednoj, praznoj tabeli. Potrebno je, nakon pisanja programa imenovati oblast ćelija koje sadrže program (komanda **/Range Name Create**) kad i da je snimite na disk. Nakon toga izlazite iz 1-2-3 i iz DOS-a pozivate program za kompilaciju AT-MAKE. Uz nju navodite i ime tabele koja sadrži program. Atmake će napraviti datoteku tipa ADN sa imenom kao kod tabele. To što ste dobili je vaša @funkcija koju možete koristiti ako je učitate kao sve ostale dodatne (*Add in*) programe.

Primeri koje sam vam pokazao su stvarno najjednostavniji kojih sam se mogao setiti. Komplikovaniji račun zahteva i komplikovaniji program. Pisanje programa bez mogućnosti za dibaging je prilično teška stvar. *3-2-1 GOSUB* je za to namenio dve specijalne @funkcije: @PRINT i @DEBUG. Pomoću njih je moguće proveriti program pre kompilacije, ali sve je to daleko od „pravih" programskih jezika. Programiranje u *3-2-1 GOSUB* nije jednostavna stvar, jer ni okruženje 1-2-3 nije predviđeno za to. No, mogu se uraditi zaista lepe i korisne @funkcije koje će smanjiti komplikovanost tabele, povećati brzinu rekalkulacije i smanjiti potrošnju memorije. Zgodno je i to što se uz program dobija i par desetina gotovih funkcija različite namene, spremnih za upotrebu.

3-2-1 GOSUB mi se prilično dopada. Uz malo odije uputstvo, uštedelo bi se dosta u vremenu, jer bi moglo pre da se počne sa poslom. Mislim da bi dosta korisnika 1-2-3 moglo da upotrebi mogućnosti koje ovaj program nudi. Posebno oni, ako ih ima, koji prave aplikacije za druge korisnike. Ako ste jedan od njih, preporučujem vam *3-2-1 GOSUB*.

WHAT'S BEST! Personal Ver. 1.5

Proizvođač: Lindo Systems Inc.
1415 N Dayton Ave
Chicago, IL 60622, USA
Tel: 800-441-2378, 312-871-2524
Cena: od 150 $ (400 varijabli) do 5000 $ (32000 varijabli)
Memorija: zauzima 256 K RAM-a
Okruženja: *Lotus 1-2-3* verzije 2.x, 3.x, *QuattroPro* verzija 2.0, DOS 2.0 ili veći

Skoro mi se nije desilo da pišem o ovako nečemu. Dobio sam program koji tretira materiju sa kojom do sada nisam imao kontakta, niti imam za nju afiniteta. Čuo sam da *What's Best!* spada u najbolje programe koji pomažu u rešavanju problema linearnog programiranja a funkcioniše u 1-2-3 okruženju. U prodaji je još od 1986. godine, kada ga je PC Magazine proglasio za vrhunski proizvod. U toku pisanja prikaza *Add in* programa za 1-2-3 malo je falilo pa da...

Na slici 1. je polazna tabela za optimizaciju. Šta se... Za ovo kratko vreme koje sam proveo uz *What's Best!* imao sam mogućnosti da se lepo poigram, zabavim i divim ljudima koji su sve to osmislili. Naime, uopšte se ne radi o programu koji se bavi linearnim programiranjem u nekom krutom, možda, matematičkom ili programerskom smislu. Radi se o programu koji se bavi izuzetno korisnom problematikom iz sveta biznisa. Verujte mi, mnogi naši privrednici, menadžeri nisu ni svesni da su mnoga rešenja njihovih problema, nad kojima lupaju glavu danima, na dohvat ruke, takoreći u sekundi. Prisetio sam se, tokom rada, nekih predavanja iz „Optimizacije" sa FON-a. Mogu da zamislim od kakve bi pomoći studentima i realizatorima bio ovaj program.

Za razliku od svih ostalih *Add in*-ova



What's Best! je jedini zapakovan kao „pravi" program u lepu plastificiranu kutiju. Program se nalazi na po dve diskete 5.25" i 3.5". Tu je i registraciona kartica. Najlepše od svega je uputstvo za upotrebu. Ne znam koliko ima stranica jer su one obeležene po poglavljima, ali je jako kvalitetno, pitko i jednostavno uvodi korisnike u mogućnosti programa.

Instalacija je, u principu, jednostavna. Program WBINST će iskopirati potrebne datoteke u 1-2-3 poddirektorijum. Ako radite sa verzijama 2.x, onda jednu disketu, a ako imate verziju 3.x, i drugu. Dakle, *What's Best!* radi sa svim verzijama 1-2-3, ali i sa *QuattroPro* verzija 2.0. Pre instalacije proverite da li ima dovoljno mesta na disku, a u toku instalacije obavezno prihvatite prenos tutorial datoteka jer će vam koristiti. Uz program dobijate i *Add in* funkciju @SUMPRODUCT koja znatno olakšava i ubrzava rad.

Pri pokretanju programa bitno je da startujete 1-2-3 kroz WB.EXE datoteku i da se *What's Best!* automatski učitava u 1-2-3.

Dakle, već sam rekao da se ovaj program može dobro iskoristiti za rešavanje pravih životnih problema u biznisu. Na primer, to su istraživanja kako ostvariti najveću zaradu u proizvodnji koristeći raspoložive zalihe, planiranje angažovanja radne snage prema potrebama, optimalni troškovi transporta proizvoda na određene lokacije ili, na primer, proizvodnja farbe sa više komponenti po minimalnoj ceni.

What's Best! Personal Version omogućuje upotrebu 400 varijabli i 200 ograničenja u maksimalno 800 numeričkih ćelija. Postoji nekoliko verzija ovog programa za različite računare, tako da najbolji i najskuplji proizvod omogućuje rad sa 32000 vrijabli.

Pokušaću na jednom jednostavnom primeru da vas zainteresujem za ovaj program i samo vam nagovestim njegove, čini mi se, velike mogućnosti.

Primer spada u grupu „Product Mix" problema, odnosno, kroz njega ću pokušati da vas uputim u način rešavanja problema ostvarivanja najveće zarade na već uvedenim proizvodima uz proizvodnju sa materijalom sa postojećih zaliha.

Na slici 1. je polazna tabela za optimizaciju. Šta se tu vidi? Imamo proizvodnju pet proizvoda koji se zovu 1, 2, 3, 4, 5. Znamo zaradu koju ostvarujemo na svakom od njih. Isto tako znamo i sirovinski sastav, odnosno komponente svakog proizvoda. U tabeli se vidi koliko jedinica svake komponente treba da bi se proizveo određeni proizvod. Poznate su i zaliha komponenti za proizvodnju svih proizvoda. Pitanje je: koji proizvod proizvoditi od komponenti sa zaliha, da bi se ostvarila maksimalna zarada?

Odgovor na ovo pitanje je čista matematika koju većina nas nije u stanju bez muke da izvede. *What's Best!* traži samo da se označe ćelije u tabeli koje učestvuju u računu, a optimizaciju, odnosno odgovor na naše pitanje pravi sam. Tu se srećemo sa tri pojma:

- **Adjustable cells** (podružljive promenljive ćelije) koje u ovom primeru treba da pokažu koliko kojih proizvoda treba napraviti da bi se najviše zaradilo.
- „**Best**" ćelija, u našem primeru najveća zarada, koja pokazuje koliko možemo maksimalno zaraditi proizvodeći proizvode iz Adjustable cells.
- **Constraints** (ograničenja) u ovom primeru pokazuju da iskorišćenje zaliha mora biti manje ili jednako količini materijala na zalihama. Stanje zaliha na kraju optimizacije ne sme biti negativno.

Na slici 2, posle optimizacije, vidi se da ćemo najveću zaradu ostvariti praveći proizvode **1** (20 kom) i 5 (10 kom), i da će nam pri tome ostati najmanje zaliha. Možete probati istu stvar kroz **What if?** iz 1-2-3, ali nećete dobiti tako dobar rezultat. Istovremeno, u tabeli se vidi koliko je zaliha iskorišćeno, koliko je toga bilo pre optimizacije i šta je ostalo na kraju.

Sada se može postaviti pitanje: šta ako smo imali ugovorene obaveze isporuke nekog od proizvoda koji ne ulazi u one koji donose najveću zaradu? *What's Best!* nudi i tu rešenje kroz takozvane dvostruke vrednosti (*Dual Values*) koje će tačno odrediti gubitke koje ćemo imati u takvom slučaju!

Ovaj i slični problemi mogu se pojaviti u raznim varijantama, sa različitim brojem promenljivih veličina. *What's Best!* može sa njima i sa gomilom drugih da izađe na kraj. Pade mi na pamet jedna mogućnost primene. Zamislite samo koliko kamiona za prevoz hleba svako jutro jurі po Beogradu. Da li svaki od njih ima optimalnu putanju kretanja? Da li se pri određivanju te putanje koriste podaci o fizičkom stanju vozila, potrošnji goriva, ulja i slično? Možda neko vozilo mora i dva ili više puta da razvozi hleb. Da li je to kamion koji će, pri tome, napraviti najmanje troškova? Dakle, pitanja ima dosta, a u traženju pravih odgovora sigurno može pomoći *What's Best!*

Program nudi još veliki broj mogućnosti. Meni ostaje da se nadam da sam uspeo da vas bar zainteresujem za ovakvu vrstu programa. Nisam imao dovoljno vremena da ga još detaljnije, i na konkretnim primerima, testiram, ali lepo sam se zabavio. Moj je prvi utisak da je reč o izuzetnom programu koji tretira probleme na čijim rešenjima direktori nepotrebno gube dane. Veoma mi se dopada!

What-If Analyst, Premium Edition

Proizvođač: Frontline Systems Inc.
140 University Avenue, Suite 100
Palo Alto, California 94301, USA
Tel: 415-329-6877
Cena: oko 100 $
Memorija: zauzima 60 K RAM-a
Okruženje: *Lotus 1-2-3* verzije 2.x, 3.x, *Symphony* sve verzije, DOS 2.0 ili veći ili OS/2 (za 123/G)

Ovaj program se proizvodi u dve varijante: *What If Analyst* (50 $) i *Premium Edition* (100 $). Oba su namenjena rešavanjima problema kroz linearne i nelinearne jednačine. *What If Analyst* rešava jednu jednačinu sa jednom nepoznatom, odnosno pronalazi jednu veličinu koja zadovoljava jednačinu. *Premium Edition*



rešava nekoliko nelinearnih jednačina, maksimum 10 sa 10 nepoznatih

What-if Analyst, Premium Edition se isporučuje na dve diskete 5.25" (360 K) ili na jednoj od 3.5" (720 K), koja se isporučuje na poseban zahtev. Uz diskete ide i uputstvo za upotrebu. Knjižica je mala, ali jasna i precizna, tako da korisnik neće imati problema kod upotrebe.

Instalacija se sastoji od kopiranja sadržaja disketa u 123 podirektorijum. U zavisnosti od okruženja (verzije 1-2-3 ili *Symphony*, DOS ili OS/2), kopiraju se određene datoteke, pedantno navedene u uputstvu za rad. Program nije zaštićen.



What-if Analyst: Linearni problem pre...



...i posle rešenja

What-if Analyst može pomoći u rešavanju mnogih problema. Klasični američki primer je traženje odgovora na pitanje po kojoj ceni treba prodati proizvod da bi se zaradila određena suma novca, a da se, pri tome, uzmu u obzir troškovi oko promocije (reklame) ili stopa rasta prodaje. Do rezultata bi mogli doći i kroz tabelu mnogim probama, ali je pitanje da li bi ikad našli pravu, najbolje vrednosti i za koje vreme.
U svojoj funkciji, u nekim elementima, je dosta sličan sa već spomenutim *What's Best!* Program se učitava kao i svaki drugi *Add-in*. Ima jednostavan meni, potpuno u Lotusovom stilu. Ključna komande su **Whatif_Cells** i **Result_Cells**. Prvom se određuje jedna ili više celija sa vrednostima koje istražujete u odnosu na jednu ili više celija sa vrednostima koje zadajete **Result_Cells** komandom. Evo jednog jednostavnog primera s kojim se možete poigrati.

Na slici vidite tabelu u kojoj možete eksperimentisati menjajući određene vrednosti da vidite kako se kreću druge, koje zavise od onih koje menjate. Predviđeno je da rast prodaje bude 10% mesečno, a da je nabavna cena robe koja se prodaje 45% od prodajne (trgovci i komercijalisti sigurno bolje znaju pravu terminologiju). Od pazara se odbijaju troškovi za robu, redovni i vanredni troškovi. Ostaje čista zarada. Pomoću *Analyst*-a možete istraživati, na primer, kolika prodaja mora da bude u tromesečju da bi ukupna zarada bila, recimo, 300.

Postupak je vrlo jednostavan. Pozovete *Analyst* komandom **Whatif_Cells** označite celiju B7 (prodaja), a komandom **Result_Cells** označite ćeliju sa ukupnom zaradom (E15) i pri tome zadate vrednost 300. Nakon toga sledi komanda **Go**, da se račun izvrši. Program je dosta brz, tako da se, kod ovakvih zahteva, izvršava trenutno. U tabeli se vide promenjene vrednosti. Ako vam ne odgovaraju, ne morate ih prihvatiti. Navedeni problem je linearnog tipa i na slikama možete videti tabelu pre i posle akcije.

Slično je pitanje promene procene rasta prodaje, da bi se ostvarila određena zarada, mada ovaj problem spada u nelinearne. Na slikama možete videti tabele pre i posle rešavanja ovog nelinearnog problema sa jednom promenljivom.

Nešto komplikovaniji slučaj je ako hoćemo da pratimo promenu i prodaje i fiksnih troškova (**Whatif_Cells**), a da pri tome ostvarimo ukupnu prodaju od, na primer, 900 i zaradu od 300 (**Result_Cells**). Najteži tip problema su oni nelinearni sa više promenljivih. Na primer, ako **what-if** ćelije sadrže i jedinica proizvoda i cene, koje kad se pomnože čine prihod, iznos za koji će zarada porasti kada dignete cene takođe zavisi od broja jedinica proizvoda.

U principu, ako hoćete jedno jedinstveno rešenje problema, morate imati jednak broj **What-if** i **Result** ćelija. Ako imate više **What-if** od **Result** ćelija, rezultat će biti jedan od više mogućih. Obrnuto, više **Result** od **What-if** ćelija obično ne daje nikakvo rešenje.

Možda će, u ova teška vremena, nekoga zainteresovati ovakav dodatni program za *Lotus 1-2-3*. Mislim da bi mnogim malim i velikim privrednicima pomogao kod donošenja odluka. Obzirom na razumnu cenu i pogodnosti koje nudi, ovaj program može biti jedno od rešenja za pomoć u poslu.

What-If Solver

Proizvođač: Frontline Systems Inc.
140 University Avenue, Suite 100
Palo Alto, California 94301, USA
Tel: 415/329-6877
Cena: oko 200 $
Memorija: zauzima 512 K RAM-a
Okruženje: *Lotus 1-2-3* verzije 2.x, 3.x, *Symphony* sve verzije i DOS 2.0 ili veći

I ovaj dodatni program za 1-2-3 pomaže u rešavanju linearnih problema, kao i nalaženju minimalnih i maksimalnih vrednosti nelinearnih funkcija sa maksimalno 40 varijabli i 20 uslovnih ograničenja. Osim toga, i varijable se mogu ograničiti gornjom i donjom granicom.

What If Solver se isporučuje na dve diskete 5.25" (360 K) i na jednoj od 3.5" (720 K). Uz diskete ide i uputstvo za upotrebu. Izgleda da je uobičajeno za Frontline Systems da daje kratka, ali prilično jasna i precizna uputstva za upotrebu, tako da se korisnik lako snalazi.



Instalacija se sastoji od kopiranja sadržaja diskete u 123 podirektorijum. U zavisnosti od okruženja (verzije 1-2-3 ili *Symphony*, DOS ili OS/2), kopiraju se određene datoteke, pedantno navedene u uputstvu za rad. Program nije zaštićen.

U *What If Solver*-u možete raditi slične poslove kao u Lotusu 1-2-3/G za OS/2. Mnogi korisnici 1-2-3 za DOS su žalili što nisu u mogućnosti da ih koriste. Ovaj *Add-in* uspostavlja ravnotežu.

What-if (šta-ako) anlize se odlično primenjuju u biznisu, ali ograničenja u primeni, praktično, nema. Kroz klasične What-if komande iz 1-2-3 možete primeniti za, na primer, rešavanje problema: koliko para ćemo zaraditi ako dignemo cenu našeg proizvoda za 10 dinara? U *What If Solver*-u pitanje bi moglo biti: koliku cenu proizvoda da odredimo da bismo zaradili 100000 dinara? Ili, koja cena proizvoda maksimizira zaradu? Ili konkretnije, kako da BIGZ minimizira troškove slanja „Računara" iz svoje kuće do krajnjeg korisnika – pretplatnika? Odgovori na ovakva i slična pitanja mogu pojedincima i firmama uštedeti velike pare. U *Solveru* možete koristiti i makro komande i procedure što svakako ubrzava rad i čini ga komfornijim. Inače rad je dosta sličan onome u *What's Best!* i *What If Analyst* programima. S toga i nisam bio posebno zainteresovan da *Solver* probam na nekom primeru.

UMESTO ZAKLJUČKA

Ako ste imali strpljenja da pročitate ovaj članak, znači da ste korisnik 1-2-3, *Symphony* ili nekog drugog *spreadsheet* programa, ili bar interesent. Namera mi je bila da vas upoznam sa malim delom sveta dodatnih programa za Lotusove tabele. Za razliku od drugih proizvođača tabelarnih programa, Lotusova strategija razvoja je nešto drugačija. U glavni proizvod, znači 1-2-3, ubacuju se samo one mogućnosti koja doprinose efikasnijem i kvalitetnijem radu, a da se pri tome zadrži maksimalna jednostavnost i brzina rada. Masa ostalih većih i manjih poboljšanja u najrazličitijim oblastima rada ostvaruje se kroz dodatne programe. Njih proizvodi i Lotus i najveći broj nezavisne softverske kompanije. Kada se ima u vidu da korisnici 1-2-3 čine najbrojniju grupu među svim korisnicima svih gotovih programa na svetu, onda je jasno i da sve što je u vezi sa tim može biti dobar biznis. Primera radi, nedavno sam pročitao procenu da je u USA, samo na obuku za rad sa 1-2-3, do sada potrošeno par milijardi dolara!!! Dokaz za to, i pored sve veće konkurencije Lotusu, je rastuće tržište *Add-in* programima za 1-2-3. Neke kompanije su napravile prave hitove za 1-2-3 i, naravno, dobar biznis.

PROFESIONALNI RAZVOJNI SISTEM

Danas, kada broj dBASE klonova iz dana u dan sve više raste, nuditi programerski alat a ne ponudit dBASE kompatibilnost ili ne biti jedno od velikih imena svjetske softverske scene, ravno je tržišnom samoubistvu. Međutim, jedna softverska kuća je rešila da krene svojim putem, nudeci tržištu nešto više od programskog jezika.

Faruk Telibećirović

Clarion Professional Developer je proizvod američke softverska kuće Clarion Software Corporation. Clarion Software je oformljen 1982. godine kao Barrington Systems, a osnivač je Bruce D. Barrington. Paket koji je prethodio *Professional Developeru* se zvao *Personal Developer* i bio je namjenjen instant programiranju jednostavnih aplikacija bez potrebe za generisanjem .EXE koda. Današnja verzija 2.1 *Clarion Professional Developera* (u daljem tekstu *Clarion*) je već sasvim zrela skupina programskih alata za razvoj profesionalnog aplikativnog softvera.

U NEKOLIKO...

Clarion spada u takozvane DBMS (*DataBase Management Systems*) pakete poput *Paradox, DataEase, R:Base* itd. Cijena paketa je izvorno 845 $, a od američkih dilera se može nabaviti i jeftinije (za 520 $ kod The Programer's Shop). *Clarion* dolazi u velikoj kartonskoj kutiji u kojoj se nalazi šest knjiga, kutija sa disketama (5 kom. 5.25" HD i 6 kom. 3.5" DD) i registraciska kartica. Kartonska kutija debelih plastificiranih zidova je namijenjena drzanju literature na polici i odlično služi svojoj svrsi. Osim registracije tu je i kartica za pretplatu na *Clarion Tech Journal*, izvrstan časopis koji vrlo precizno prati svu *Clarion* problematiku, podloška za tastaturu i karton Clarion Utilities Quick Reference.

Sistemski zahtjevi za instalaciju nisu veliki: DOS 2.0 ili viši, 512 K memorije (640 K se preporučuje), 6 MB prostora na hard disku i jedan flopi drajv. Za komotan rad sa paketom bih preporučio najmanje AT mašinu, a najbolji rezultati se svakako postižu na 386/486 sa 2 ili više MB i što bržim hard diskom. Instalacija paketa se svodi na pokretanje programa INSTALL sa prve diskete, postavljanje parametara instalacije i na izmjena disketa kako procedura to zahtjeva.

Na početku procedura možemo zahtjevati instalaciju kompletnog sistema, nakon čega INSTALL sam kreira sve direktorije sa zadatog nivoa, npr. C:/JEZICI/CLARION. Osim CLARION direktorija, INSTALL kreira i poddirektorije EXAMPLES, TUTORIAL, REPORT i DATABASE u koje smješta primjere, program za učenje, izvještaje kreirane *Report Writer*-om i baze podataka za izvještaje. Korisniku je ostavljeno na volju da sam kreira potrebne direktorije i da u njima vrši razvoj svojih aplikacija. *Clarion* je dovoljno inteligentan da pamti putleve specifične za pojedine aplikacije i automatski ih predlaže korisniku.

Pored PATH-a, *Clarion* štuje još dvije promjenjive iz *environment*-a: LIB i OBJ (iako često sasvim dobro radi i bez njih). Veličine za BUFFERS i FILES su specifične za pojedine korisnike (normalno BUFFERS=20 i FILES=40) zbog sve češće upotrebe programa za keširanje. U CLARION direktoriju se nalazi osamdesetak datoteka, od čega su najveći dio *Clarion* alati i njihovi helpovi. Svi alati se mogu pozivati iz DOS prompta, kao i iz *Clarion* školjke (CLARION.COM program).

Izvanredan dio *Clarion* paketa je *Tutorial* program za učenje. *Tutorial* je podijeljen na segmente koji pokrivaju sve *Clarion* alate, a najzanimljiviji su *Designer* i *Clarion Language*. Koristeći *Tutorial* moguće je ostvariti početne korake u *Clarionu* bez napornog listanja uputstva. Za kasniji profesionalni rad uputstvo je kao i kod drugih softverskih alata neizbježno.

Clarion literatura se sastoji iz šest tematski pisanih knjiga: *Getting Started, Utilities Guide, Language Reference, Annotated Examples, Report Writer* i priloga *Applications & Resources*. Format knjiga je došla neobičan (valjda zato da bi se otežalo fotokopiranje), a kvalitet pisanog teksta je takav da je razumljiv i onima koji nisu najbolje ovladali engleskim jezikom.

Za početak rada u *Clarion*-u je najbolje prepravit datoteku OPTIONS.CTL (koristeći OPTIONS.PRO program) tako da podržava YU sortni raspored i upper-lower konverziju. Time dobijamo mogućnost generisanja aplikacija prilagođenih našim potrebama bez pisanja nepotrebnih sort rutina.

Za sve one koji se profesionalno bave programiranjem *Clarion* je idealan alat. Distribucija .EXE verzije aplikacija ne podliježe ograničenjima. Takođe se uz *Clarion* odbija Run-Time sistem čija je distribucija uz aplikacije ovlaštena od strane Clarion Software-a. To omogućava da, ukoliko na jednoj instalaciji imate više aplikacija, ne morate generisati višestruke (inače relativno duge) .EXE datoteke. Osim toga dozvoljena je distribucija alata za konverzija datoteka i *Clarion Scanner* alata.

Početna dužina *Clarion* programa je relativno velika. *Clarion* program koji izvršava samo jednu naredbu (ispisuje „zdravo" na ekran) je dug 138 K. To je zbog „ugradnje" funkcija iz biblioteke u izvršni program. Međutim, dalji rast veličine programa je mnogo umjereniji i iznosi u prosjeku 5 do 15 K po proceduri. Primjera radi, jedna vrlo kompleksna aplikacija za podršku finansijskog poslovanja srednjih i većih preduzeća (sadrži 117 procedura) ima u .EXE verziji nešto preko 900 K koda.

O kompatibilnosti nema potrebe mnogo govoriti. Ukratko, *Clarion* je 100 % DOS kompatibilan i koristio sam ga na skoro svim DOS platformama (8086, 288, 386 i 486). Sasvim normalno radi i na svim verzijama DOS-a (3.30, 4.01 i 5.00 isprobano). Pošto u firmi u kojoj radim pored PC tehnike imamo i IBM AS/400, pojavila se potreba za povezivanjem aplikacija na dislociranim stanicama sa hostom u Sarajevu. I ovde je *Clarion*, ostvarivši saradnju sa IBM PC SUPPORT softverom, položio ispit kompatibilnosti.

Za *Clarion* postoji veliki broj dodataka, direktno od Clarion Software ili pak nezavisnih proizvođača. Zajednički naziv za jednu vrstu tih dodataka je LEM (*Language Extension Module*) i nude korisniku proširenje mogućnosti *Clarion* jezika potpuno novim funkcijama. Jedan od vrlo zanimljivih LEM ova je i *Comunications LEM* koji pruža potpunu kontrolu nad serijskim interfejsom PC računara, dozvoljavajući pritom pisanje aplikacija u koje će recimo biti ugrađena podrška za modemski prenos podataka. U nastavku teksta opisaću rad pojedinih modula *Clarion Professional Developera*. Logično je očekivati da prvi od njih bude



CLARION DESIGNER

Alat po kome je *Clarion* stekao svoju reputaciju je svakako *Dizajner* – generator aplikacija po principu crtanja i bojenja (nekom od 16 boja iz palete) željenih struktura. *Dizajner* nas oslobađa mukotrpnog rada oko kreiranja datoteka, ekrana i izvještaja. Dosadašnja metoda „proba – ispravka – proba" uzima bar deset puta više vremena i uz to daje mnogo siromašnije rezultate. Jedini efikasan način rada je bio da koristeći **cut paste** opcije editora iskopiramo neki

od već odrađenih ekrana i izmjenimo samo ono najnužnije. To je razlog što svi programi jednog autora počinju vremenom ličiti.

Dizajneru pristupamo na tri načina: sa CDES iz DOS prompta, iz *Clarion* šela, ili pritiskajući *Hotkey* kombinaciju F3, odnosno Shift-F3 bilo gdje unutar *Clarion* okoline (sa usnimavanjem aktivne aplikacije). U prvom i drugom slučaju *Dizajner* nas pita za naziv aplikacije sa kojom želimo da radimo, naziv model datoteke po kojoj se generiše izvorni kod, naziv help datoteke i naziv bazne procedure. Ukoliko aplikacija (.APP datoteka) ne postoji u odabranom direktoriju, onda nas *Dizajner* obavještava da će aplikacija sa zadatim imenom biti kreirane. U trećem slučaju se automatski usnimava posljednja aktivna aplikacija, tako da je ubrzan prelazak iz drugih *Clarion* alata tokom razvoja aplikacije.

Po startovanju, *Dizajner* nas pita da li želimo tzv. **Quick Start** opciju gdje na početku samo ubrzano definišemo datoteke i pristupne puteve, a *Dizajner* automatski generiše sve potrebne menije, tabele i forme za opsluživanje tih datoteka. Ova opcija je vrlo zanimljiva svima onima koji bi da kreiraju male baze podataka, bez puno datoteka i bez nekih složenijih relacija unutar baze. Idealno za male biblioteke, video klubove ili na primjer evidenciju osnovnih sredstava u nekom malom preduzeću. Sve automatikom generisane procedure i datoteke možemo ručno modifikovati, brisati ili pak dodavati nove strukture da bismo zadovoljili neke druge zahtjeve koji se postavljaju pred aplikaciju. Dakako, automatiku je moguće izbjeći izborom opcije „No" na pitanje „Use Quick Start?". U tom slučaju sami definišemo datoteke i procedure, ili koristimo Ctrl-G kombinaciju tastera za **browse-copy** definicije datoteke ili procedure iz neke druge aplikacije.



Slika 1 Osnovni *Clarion* meni. Iz ovog menija moguće je pristupiti svim *Clarion* alatima

Osnovni ekran *Dizajnera* je podijeljen na dva dijela: lijevi uži dio je za datoteke, a desni prikazuje stablo procedura. Kretanje po ekranu je olakšano upotrebom kursorskih tastera, kao i prelazak iz jednog dijela ekrana u drugi. Pritiskom na taster **Insert** dodajemo novu datoteku ili proceduru, sa **Delete** brišemo, a sa **Enter** ukazujemo da želimo mijenjati istu. Filozofiju tastera **Ins**, **Del** i **Enter** koriste i generisane aplikacije (ovo je moguće promijeniti u model datoteci), što je vrlo korisno zbog mogućnosti ostvarivanja jedinstvenog korisničkog interfejsa.

Definisanje datoteka se svodi na:

1) Zadavanje osnovnih parametara datoteke: naziv, automatsko kreiranje, popunjavanje obrisanih slogova, opciono kriptovanje datoteke.

2) Opisivanje polja koja se nalaze u datoteci: tip i dužinu polja, način unosa i ispisa, kao i Help ID potreban da bi se uz polje mogao definisati i help ekran.

3) Opisivanje pristupnih puteva.

Clarion podržava dvije vrste pristupnih puteva, a to su KEY i INDEX. Prvi se permanentno osvježavaju i omogućavaju zaštićeni pristupni put u svako vrijeme, dok je druge potrebno „izgraditi" prije upotrebe. Ključevi mogu biti označeni kao jedinstveni, tako da pokušaj unosa duplih podataka izaziva dojavu greške.

Procedure se definišu na sljedeći način:



Slika 2 Bazni ekran *Dizajnera*

1) Odaberemo vrstu procedure. Procedure mogu biti definisane kao meniji, tabele, forme, izvještaji i ostale.

2) Popunjavamo parametre procedura. To su **Help** procedure, **Hot Key** procedure, **Update** procedura, kao i opciono naziv Modela procedure.

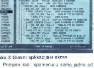
3) Pristupamo dizajniranju-crtanju ekrana ili izvještaja.

Sve što unutar *Dizajnera* opišemo ili nacrtamo, kasnije biva prevedeno u izvorni kod (.CLA datoteke) i može biti dotjerivano i mijenjano. Ovaj proces se može pokrenuti globalno (za cijelu aplikaciju) ili percijalno (ova opcija se najčešće koristi za dibagiranje ili za male dorade na dijelovima aplikacije). Generisanje izvornog koda može biti preskočeno ukoliko želimo kasnije nastaviti sa radom, pa prevođenje aplikacije u tom trenutku nema smisla.

Kombinacija tastera **Shift-F7** angažuje bateriju **Designer-Compiler-Processor** i dozvoljava programeru da u vrlo kratkom roku može vidjeti rezultat svoga rada. Recimo, utvrdimo da nam se ne dopada položaj zaglavlja na faktur. Ulazimo u *Dizajner* i pozicioniramo se u traženu proceduru (report), vršimo pomjeranje zaglavlja na željenu poziciju (Ctrl-M), pritiskom na Shift-F7 pokrećemo proces. Bez naše dalje intervencije *Dizajner* generiše izvorni samo za proceduru koja je upravo mijenjana, poziva kompajler koji prevodi upravo generisan izvorni kod, zatim se automatski poziva procesor koji pokreće aplikaciju bez potrebe za linkovanjem. Programer može da pozove upravo modifikovanu proceduru, isproba je i da, ako nije zadovoljan dobivenim rezultatom, ponovi prethodni proces neograničen broj puta. Navedeni ciklus korekcija-proba može da traje između 15 i 20 sekundi po mijenjanoj proceduri, što može da varira zavisno od kompleksnosti struktura i od brzine mašine na kojoj se radi (referensa 386/33 4MB + PC-Kwik). Tako kratak ciklus omogućava znatno povećanje produktivnosti ne samo na račun brzine rada, već i zbog održavanja kontinuiteta koncentracije programera.

Model (.MDL) datoteka je šablon po kome *Dizajner* generiše izvorni kod. Način na koji je to izvedeno je dosta jednostavan. Model datoteka je najvećim dijelom napisana „običnim" *Clarion* jezikom, a tek manji dio uz upotrebu tzv. *keywords*. Namjena tih „ključnih riječi" je da označe mjesta na kojima *Dizajner* ubacuje podatke specifične za pojedine procedure i druge programske strukture. Dakako, kompleksne strukture, kao što su ekrani i izvještaji, su opisane samo jednom ključnom riječju u modelu, a *Dizajner* na tom mjestu generiše cjelovitu strukturu u *Clarion* jeziku. U sastavu model datoteka ulaze modeli procedura i struktura, koje *Clarion* prepoznaje po zvjezdici "*" ispred imena modela procedure (u daljem tekstu modela). Uz *Clarion Professional Developer* dobivamo dvije model datoteke: NETWORK.MDL i STANDARD.MDL. Kao što se može pretpostaviti, prva služi za opsluživanje mrežnih okruženja, a druga za generisanje programa koji će raditi na jednom računaru.

Način na koji su model datoteke pisane osigurava podršku najvećem broju korisničkih zahtjeva. U isporučenim model datotekama podržani su sljedeći modeli: **Menu** (rad sa menijima), **Table** (rad sa ekranskim tabelama), **Form** (rad sa maskama za unos) i **Report** (generisanje izvještaja). Iznenađujuće je činjenica na koliki broj programskih zahtjeva se može odgovoriti upotrebom nekog od gotovih modela. Procentualno izražen, ovaj broj ne pada ispod 80%, čak i kod veoma kompleksnih aplikacija (npr. fakturisanja sa automatskim kontiranjem u finansije). Iz prethodnog slijedi da je za samo 20% aplikacije potrebno pozivati editor – ostatak generiše *Dizajner* bez dodatnih intervencija editorom u izvornom kodu. Specifični programski zahtjevi se mogu ostvariti nadopunom postojećih modela, kopiranjem postojećih modela u nove pod novim imenom (što je preporučljivije) i kreiranjem vlastitih modela upotrebom potrebnih „ključnih riječi" (što je dosta teško, ali se isplati). *Dizajnerom* generisan kod prema model datoteci je kvalitetan, sa velikim brojem predviđenih opcija (neiskorištene opcije se izostavljaju), cjelovitom kontrolom potencijalnih grešaka i komentarima na kraju skoro svake programske linije.



Slika 3 Glavni aplikacijski ekran

Primjera radi, spomenuću samo jednu od mnogobrojnih prednosti ovakvog načina pisanja programa. Kod generisan iz modela predviđa automatsko otvaranje svih definisanih datoteka. Ukoliko prilikom otvaranja datoteka dođe do prijavljivanja greške, kontroliše se tip te greške. U slučaju da datoteka ne postoji, program je automatski kreira. Moguće je da program ne nađe datoteku ključeva predviđenu uz upravo otvaranu datoteku. U navedenom slučaju program kreira potrebne ključeve. Čest je slučaj da (zbog lijenosti) ove opcije programeri ne stavljaju u svoje aplikacije, ili ne očekuju da će se takve greške kad-tad pojaviti, a kada se pojave onda se krivica svaljuje na drugog. Greške ovog tipa se možda i neće pojaviti, ali se recimo nedostatak prostora na disku ili zaključanost traženog sloga pojavljuje itekako često da bismo te slučajeve ostavili nepokrivene.

CLARION EDITOR

Ako se i jedna programerska navika urezuje u svijest programera jače od programskog jezika, onda je to editor. Moć navike je učinila da



Slika 4 Dizajniranje datoteke



Slika 5 Dizajniranje polja datoteke



Slika 6 Dizajniranje *Table* procedure



Slika 7 Polje za unos u sastavu *Table* procedure

sam inače odličan editor radije zamijenio za stari malo poznat ali zato iskušan i fleksibilan HACK. Iz radoznalosti sam nabavio i *Multi-Edit Professional* ali uzalud navika ostaje navika. Inače imam kolegu vrsnog Cobol programera koji je cijeli svoj programarski vijek provao pišući u EDLINU. Da dobro ste pročitali EDLINU iz DOS a! Uzalud su bila sva moja uvjeravanja o potpunoj nefleksibilnosti i jalovosti tog editora – on je ostao pri svom. *Clarion*-ov Editor upotrebljavam za ispravke grešaka koje prijavi prevodilac ili za povremeno formatiranje ekrana i izvještaja direktno u izvornim datotekama.

Editor se sastoji iz tri dijela:

- SOURCE EDITOR za pisanje i editovanje *Clarion* izvornog koda, kao i za modifikaciju koda koji generiše *Dizajnera*.
- SCREEN FORMATER za kreiranje i formatiranje ekrana kao i za modifikovanje već postojećih ekrana.
- REPORT FORMATER za kreiranje i formatiranje izvještaja kao i za modifikovanje već postojećih izvještaja.

Editor možete pozvati direktno iz DOS-a sa CEDT ili iz *Clarion* glavnog menija izborom opcije **Editor**. Postoji opcija za poziv sa bilo kojeg mjesta unutar *Clarion* okoline pritiskom na **F5** ili **Shift F5** za usnimavanje aktivne izvorne datoteke.

CLARION HELPER

Velika boljka većine domaćih aplikacija je u nedostatku podrške korisniku tokom samog korišćenja aplikacije. Dakako riječ je o poznatim help-ovima ugrađenim u većinu programa koji su namijenjeni široj upotrebi. Isti problem se samo djelimično rješava štampanjem obimne literature uz aplikaciju, ali stalno zavirivanje u uputstvo radi recimo nalaženja prave kombinacije tastera za neku od opcija koje se često koriste može da postane mučenje za korisnika aplikacije. *Clarion Helper* sjajno rješava ovaj problem. Dovoljno je da uz polje koje definišemo unutar neke datoteke ili uz proceduru navedemo tzv. **Help ID** i kasnije upotrebom Helpera kreiramo help ekran (ili čak help meni). Tokom rada sa aplikacijom kada se nađemo na polju podržanom **Help ID**-om, pritiskom na taster **F1** (standardni help taster) dobijamo na ekranu ranije definisani help ili **help menu**. Obzirom da se helpovi čuvaju u zasebnoj HLP datoteci moguće je njihovo modifikovanje bez diranja ostatka (gotove) aplikacije. To vam dopušta da za „specijalne" korisnike posebno naglasite neke od opcija unutar programa koje stalno ponavljate (telefonom ili lično).

CLARION JEZIK

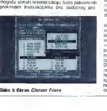
Clarion jezik je priča za sebe. Sintaksa podsjeća na *Clipper* i paskal, ali je evidentno da su neke dobre strane bejzika ugrađene u ovaj izuzetno prijatan jezik jer za uspješno programiranje ne zahtijeva od programera poznavanje RSA, DES ili nekog drugog kriptografskog algoritma (for(P(./n") R P(,,"))for(e=C e-) Znak jednakosti je i dodjeljivanje i komparacija a uskličnik označava početak komentara.

Sve promjenjive se moraju deklarisati na početku programa, ili ako je riječ o lokalnim promjenjivim, na početku procedura. Izuzetak od ovog pravila su implicitne promjenjive koje se deklarišu prilikom upotrebe. Jezik dozvoljava upotrebu eksterno deklarisanih varijabli kao i datoteka, izvještaja, ekrana.

Clarion podržava sledeće tipove promenljivih: BYTE – cjelobrojna promjenjiva u opsegu 0 do 255, SHORT – cjelobrojna u opsegu -32 768 do 32 767, LONG – cjelobrojna u opsegu -2 147 483 648 do 2 147 483 647, REAL – Intel 8087 kompatibilan format floating point promjenjive dvostruke preciznosti (14 tačnih cifara), DECIMAL – pakovana decimalna promjenjiva sa dvije cifre po bajtu i STRING – klasična karakter promjenjiva.

Od struktura su podržane gotovo sve: IF THEN ELSIF ELSE CASE OF OROF ELSE EXECUTE LOOP, LOOP TIMES, LOOP from TO BY, LOOP UNTIL, LOOP WHILE.

Ogromna snaga se nalazi u tzv. COMMITMENT kontroli ugrađenoj u sam jezik. Tako je moguće izvršiti rekonstrukciju baza pokvarenih prekinutim transakcijama bez dodatnog programiranja. Dovoljno je na početku transakcija izdati naredbu LOGOUT(**naziv logdat**) i da se od tog trenutka sve promjene nad datotekama zapisuju na posebno mjesto. Nakon uspješno završenih transakcija naredbom COMMIT brišemo log datoteku. Ukoliko je tokom izvršavanja transakcija došlo do nasilnog prekida rada računara (nestanak napajanja i sl.) tada instrukcijom ROLBACK(**naziv logdat**) ugrađenom na početak programa rekonstruišemo datoteke na stanje prije izvršene naredbe LOGOUT.



Slika 9 Ekran *Clarion Filera*



Slika 8 Bazni ekran *Kompajlera*

Datoteke se mogu otvarati ekskluzivno ili *share* za mrežni rad. Maksimalan broj otvorenih datoteka nije limitiran *Clarion*-om već DOS-om. Tako da na samostalnoj mašini iznosi 43 zbog već poznate problematike sa **Maximum Handle Count** funkcijom interapta 21H za otvaranje više od 20 datoteka. Pod mrežom se taj problem odstranjuje u *NetWare* SHELL.CFG datoteci postavljanjem FILE HANDLES=n reda. Maksimalan broj slogova nije limitiran. U stvarnosti sam radio sa bazama od 2 miliona slogova sa nešto izraženijim usporenjem obrade. Navedeni broj slogova se rijetko sreće čak i u većim preduzećima pa se može smatrati da je *Clarion* vrlo upotrebljiv za poslovnu obradu podataka. Odlično se slaže sa C-om, asemblerom, paskalom i u uputstvu je dokumentovan njihov način povezivanja. *Clarion* jezik podržava rad sa overlejima tako da se na taj način omogućava pisanje programa dužih od 640 K.

Preko datoteke OPTIONS.CTL moguće je definisati kodni raspored po želji i podesiti konverziju **upper-lower** (dakle YU redoslijed bez dodatnih sortova) tako da su od tog trenutka sva setanja po bazi u stilu ABCČĆ ... a ne ili ABCDEF ... Takođe je moguće prevesti sve poruke o greškama, čak i one kritične, kao i nazive mjeseca, oznaku AM/PM i još neke opcije. *Clarion* je pisan korišćenjem isključivo DOS poziva tako da je kompatibilnost maksimalno moguća. Brzina je donekle rezultat prethodnog, ta je po mojim usporedbama *Clarion* sporiji od *Clippera* za oko 15-20%, što po meni nije toliko značajno (lično bih pristao i na lošiji odnos).



Slika 10 Ilustracija iz jedne aplikacije u *Clarion*-u

REPORT WRITER

Clarion Report Writer (ili krace *Reporter*) je dodatak koji se pojavio tek od verzije 2.1 *Clarion Professional Developera*. Njegova funkcija je da olakša generisanje štampanih i ekranskih izvještaja iz postojećih baza podataka. Ova mogućnost je idealna za krajnjeg korisnika aplikacije koji bez programiranja može da generiše veliki broj potrebnih izvještaja što je do sada zahtijevalo dodatno angažovanje programera. Da bi mogli kreirati izvještaj potrebno je da prije toga definišemo bazu podataka. Baza se definiše tako što biramo datoteke sa kojima operišemo, zatim određujemo koja polja se koriste za vezu među datotekama i na kraju predlažemo načine ispisa za pojedina polja iz tih datoteka. Nakon postupka definisanja baze podataka prilazimo kreiranju izvještaja: postavljamo polja na željene pozicije, određujemo sortni redosljed, filtere, promjenjiva polja i druge opcije. Nakon kreiranja izvještaj možemo generisati na ekran, štampač ili u datoteku. Generisanje izvještaja se može pokrenuti iz DOS prompta koristeći RRUN.EXE program čija je distribucija dozvoljena uz gotove aplikacije. *Reporter* radi i sa formatima podataka različitim od *Clarion*-a: podržava ASCII, dBASE II, dBASE III i DIF.

CLARION COMPILER

Clarion kompajler spada u grupu jednoprolaznih kompajlera za više programske jezike orijentisane obradi baza podataka. Kompajler prevodi .CLA datoteke u međukod (.PRO datoteke) koji je moguće izvršavati pod *Procesorom*. Kasnije se *Translatorom* vrši konverzija .PRO u .OBJ i pozivanje linkera za generisanje .EXE verzije. Interesantna je opcija da kompajler sam prepoznaje da li su neke od struktura mijenjane od prethodnog kompajliranja i shodno tome dozvoljava parcijalno prevođenje ili insistira na kompletnom prelazu svih programskih modula.

CLARION PROCESSOR

Procesor omogućava pokretanje, testiranje i dibagiranje aplikacija bez startovanja procesa linkovanja. Da bi se vrijeme razvoja aplikacije svelo na najmanju moguću mjeru, potrebno je obezbijediti brz i efikasan proces testiranja i dibagiranja. Dosadašnji postupci kompajliranja, zatim linkovanja pa testiranja nisu pružili baš najbolje rezultate. Linkovanje je proces koji kod većih aplikacija zna dekako potrajati, zato se Clarion Software dosjetio da je moguće jednu verijantu .OBJ koda (.PRO kod) usnimiti u memoriju zajedno sa svim funkcijama iz biblioteka i kao takvu startovati. Ponašanje ovako pokrenutih programa se ne razlikuje od programa prevedenih u .EXE oblik. Ovo svojstvo je iskorišteno još i za opciju distribucije *Run Time* modula CRUN.EXE za platforme sa višestrukim *Clarion* aplikacijama, a sve to radi uštede memorijskog prostora na disku od stotinjak K po aplikaciji (veličina .PRO koda je više stotine puta manja od konačne .EXE verzije).

Dibager ugrađen u *Procesor* spada u red solidnih dibagera koji pružaju mogućnost postavljanja prekidnih tačaka, izvršavanja korak po korak, uvida u izvorni kod, uvida u stanje sistemskih resursa, pregled otvorenih datoteka, način otvaranje datoteke i još čitav niz interesantnih opcija.

CLARION TRANSLATOR

Translator je alat koji se brine za konverziju koda prevedenog od strane kompajlera u .OBJ a opciono i u .EXE oblik. Korištenje *Translatora* se obično svodi na generisanje konačne verzije aplikacije (.EXE koda). Pošto *Clarion* od svoje verzije 2.0 upotrebljava RTLink koji je poznat korisnicima novog *Clipper*-a 5.0, sva potrebne parametre za linker pripreme Translator, oslobodivši korisnika od „nagađanja" potrebnih parametara.

CLARION CROSSREFER

Crossrefer služi za generisanje izvještaje gdje se koja promjenjiva upotrebljeva. Ova opcija je jako korisna za dibagiranje kao i za prepravke davno napisanih aplikacija. Takođe se može upotrebiti za razradu strategije ovarleja. *Crossrefer* generiše .XRF datoteku sa svim kodovima potrebnim za dobijanje jasne i pregledne štampe, tako da je dovoljno datoteku uputiti na štampač u izvornom obliku.

CLARION DIRECTOR

Director predstavlja jednu od mnogobrojnih verijanti DOS školjki. Omogućava kopiranje, brisanje, promjenu imena, štampu i druge opcije. Pošto radije upotrebljavam univerzalnije alate, tamo gdje mogu koristim PCSHELL iz PCTOOLS-a.

CLARION TAILOR

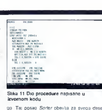
Tailor služi za prilagođavanje *Clarion* okoline zahtjevima korisnika. Kontrolišu se: sjenke prozora, boja, upozoravajući pisak, video timeout funkcija (prenosi se na generisanu aplikaciju), opcije kompajlera i *Dizajnera*.

CLARION FILER

Programiranje u drugim jezicima obično zahtijeva pisanje servisnih rutina kada se želi promjene nad strukturama podataka. *Filer* je jedan od *Clarion* alata koji olakšavaju promjene struktura datoteka kao i kreiranje novih datoteka. Referensa kojom se pritom služi je izvorni kod u kome je datoteka opisana. Moguće je promijeniti raspored polja, usmjeriti jedno polje u drugo, dodavati nova polja, prepisivati iz datoteke u datoteku. Svaki program kreiran od strane *Dizajnera* ima mogućnost kreiranja datoteke ukoliko ista ne postoji u radnom direktoriju. Pošto ova opcija zahtjeva nešto radne memorije i donekle ugrožava zaštitu programa, alternativa je da se datoteke kreiraju uz pomoć *Filer*-a.

CLARION SORTER

Sorter spada u grupu alata za uređivanje datoteka i manipulisanje njima. Ponuđene su opcije za izbor ulazne i izlazne datoteke, ključa za sortiranje, kompresiju datoteke, pakovanje brisanih slogova i odstranjivanje ključeva iz strukture datoteke. *Sorter* nudi najbržu viđenu opciju izgradnje ključeva. Baza podataka sa telefonskim brojevima pretplatnog područja Sarajevo (071) ima 133009 slogova (stanje iz maja mjeseca) i ključevi su organizovani po svim podacima (telefonski broj, adresa, ime i prezime, mjesto), te je za pretpostaviti da izgradnje svih ključeva za ovu bazu može potrajati dosta dugo. Taj posao *Sorter* obavlja za svega deset minuta (386/33 4MB RAM).



Slika 11 Dio procedure napisane u izvornom kodu

CLARION SCANNER

Za šetanje po *Clarion* datotekama, direktno ispravljanje podataka kao i za dodavanja i brisanje slogova služi *Scanner*. Za pretraživanje baza tu je **Search & Replace** opcija. Format prikaza vrijednosti u poljima je moguće podesiti na željeni oblik, a takođe i grupisati interesantne polja tako da se nalaze zajedno na ekranu. *Scanner* se može definisati i kao podskup funkcija jednog DBU-a za dBASE i po tome donekle zaostaje za sličnim paketima.

CLARION CONVERTER

Pošto Clarion Software sebe ne vidi kao jedinog ponuđača na tržištu, logično je očekivati i ponudu alata za konverziju raznih formata datoteka. Podržan je prije svega dBASE II i III, zatim ASCII, bejzik i na kraju DIF (Lotusov format). Moguća je obostrana konverzija, kako iz „stranih" formata u *Clarion* tako i iz *Clarion*-a u druge formate. Opciono *Converter* generiše i izvorni kod u kome je opisana prevedena struktura, pa se onda taj izvorni može ugraditi u neki *Clarion* program.

...NAJTOPLIJIH REČI

Ovim bih završio prikaz *Clarion* paketa. Koristim ga već dvije godine i smatram ga odličnim izborom. *Clarion* samo olakšava posao programerima, ali ga ne degradira, već im dozvoljava da se posvete srži problema i da ne gube vrijeme na nebitnim stvarima (pisanje ekrana, kontrole grešaka i slično). Da mi neko ne bi zamjerio na neupućenosti zbog poređenja *Clarion*-a sa drugim programskim jezicima, napominjem da imem i *Clipper 5.0* (verziju 5.01 još nisam vidio), ali nikad ne bih ozbiljno prihvatio proizvod koji sa objavom nove verzije korisnicima napravi više štete nego koristi i uz to izgubi kompatibilnost sa starom verzijom (npr. *Clipper* naredba REPLACE ne podržava više blenkove među parametrima). Pod štetom mislim na silno očekivanje dugo najavljivane verzije *Clipper*-a 5.0 i na zapostavljen razvoj upravo zbog očekivanja novog alata.

Clarion sebi nije dozvolio takav luksuz od kako znam za njega. Njegove mogućnosti svakako nisu zanemarljive, tim prije što je na pomolu nova verzija 3.0, kao i verzija za UNIX. Najavljene promjene, poboljšan *Dizajner*, izbor drajvera za baze podataka (*Oracle*, *BTrieve*, dBASE i drugo) svakako opravdavaju dosadašnju *Clarion*-ovu reputaciju da se brojevi verzija ne mijenjaju tek tako. Kada je već riječ o novoj verziji, treba napomenuti da je Clarion Software još uvijek ne reklamira na sva zvona. Valjda zbog oprobane strategije prihvaćene od IBM-a: „govori o novom proizvodu tek kada je na policama prodavnica i kad je očišćen od bubica". Svaka im čast!

NAKON PREKIDA VEZE

Modemske komunikacije su još uvek daleko od savršenstva – relativno male brzine prenosa u odnoau na ono što su računari sami za sebe u stanju da urade, problemi sa šumom na vezi koji ometa komukaciju... Ipak, modemi su zarazili ceo svet, a problemi se, zahvaljujući nekim dodatnim alatima, mogu svesti na najmanju meru.

Zahvaljujući protokolima za prenos podataka (XMODEM YMODEM ZMODEM) možemo potpuno pouzdano, bez i jedne greške, preneti neograničenu količinu podataka. Zahvaljujući arhiverima (programi za kompresiju kao što su ZIP, ARC ili ARJ) prenosi podataka traju znatno kraće. Zahvaljujući MNP protokolu možemo za vreme same veze primati i čitati tekst bez grešaka. I, konačno, zahvaljujući *off line reader*-ima možemo jednostavno i uz minimalni telefonski račun uživati u razmeni poruka i diskusiji sa drugim korisnicima, slažući tako dobijene informacije u vrednu bazu podataka. Ovaj tekst je posvećen jednom programu iz ove poslednje grupe – *Sezam Off-Line Reader* (SOR) je program koji je nastao za potrebe korisnika *Sezama*, BBS sistema našeg časopisa.

DOK IMPULSI ODBROJAVAJU...

Kao i mnoge termine i pojmove iz oblasti računarstva, ni naziv *Off-Line Reader* (OLR) nema adekvatan prevod na naš jezik. Najkraće rečeno, radi se o programu koji omogućava da, nakon što ste pokupili potrebne informacije sa nekog servisa i prekinuli vezu, na miru, bez opasnosti po telefonski račun i sa punim komforom koji računar može u ovoj oblasti da pruži, pregledate primljene poruke, odgovarate na njih i vredne informacije ostavljate trajno u bazi poruka.

Postoji nekoliko problema kod on line čitanja poruka (čitanje u toku samog trajanja veze). Svakako najozbiljniji je taj što se potpuno nepotrebno troši vreme. Da bi se pročitala jedna tipična poruka od 20-tak redova, potrebno je oko pola minuta. Ako se uzme u obzir da na *Sezam* dnevno pristigne između 200 i 300 javnih poruka, od kojih će vas sigurno bar polovina zanimati, jasno je da je čak i pri redovnom svakodnevnom javljanju potrebno suviše vremena da se one pregledaju, što samo može povećati nervozu u radu kod korisnika koji zovu iz udaljenih gradova. Poruke se, naravno,

Zoran Životić

mogu i preskakati, ali je ipak potrebno da vidite bar nekliko prvih redova da biste ustanovili da li vas poruka zanima ili ne.

Sledeći problem nastaje kada zbog smetnji na telefonskim vezama deo poruke primite sa puno grešaka – tada treba prekinuti tok čitanja i vratiti se na prethodnu poruku. Slična stvar se dešava ako je u toku čitanja nekog odgovora potrebno da pogledate originalnu poruku na koju se odgovara. Tada takođe treba zastati i zadati novu komandu.

Zbog onih koji nemaju iskustva sa modemskim komunikacijama treba reći da se komunikacija sa BBS-ovima najčešće odvija u linijskom režimu. Podaci se na vaš ekran ispisuju slično kao kod matričnog štampača – kada neka informacija „pobegne" sa ekrana, nema više načina da se kursorskim tasterima vratite na nju. Ovo postaje posebno značajno ako poželite da odmah, u toku čitanja, reagujete i odgovorite na neku poruku. Ulaskom u editor za unos poruke gubite mogućnost da se ponovo podsetite sadržaja poruke na koju odgovarate. Naravno, ovaj problem se veoma lako rešava kod savremenih komunikacionih programa, koji na vašem računaru pamte ceo tekst koji ste od početka veze primili i omogućavaju vam da se na njega vratite, ali je i za ovu radnju potrebna određena snalažljivost, pogotovo što se nikada neće osloboditi osećaja da za vreme vaše „šetnje" po onome što se prethodno desilo telefonski impulsi i dalje neumoljivo odbrojavaju.

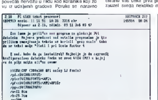
...ILI NA MIRU, KOD KUĆE

Zamislite sada ovakvu sliku – uključujete vaš računar, pokrećete poseban program, odlazite da skuvate kafu i kada se vratite, na ekranu vas čeka prva poruka. Telefon je bio zauzet svega nekoliko minuta, a vama sada ostaje da na miru pročitate poruke. Pri tome se, pritiskom na samo nekoliko tastera, za delić sekunde vraćate na bilo koju staru poruku, preskačete poruke koje vas ne zanimaju, opet samo sa nekoliko tastera ulazite u vaš omiljeni editor u kome vas čeka citat poruke na koju se spremate da odgovorite itd. Ukratko, radite na onaj način na koji ste navikli – računar reaguje trenutno na svaki vaš zahtev. Kada ste napisali sve odgovore, ostaje vam da ponovo pokrenete program, koji će sam pozvati sistem i ostaviti vaše poruke.

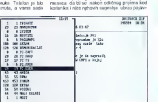
Ako vam u toku vašeg redovnog posla zatreba neka informacija, a sećate se da je o tome bilo reči u nekoj poruci, pokrenete ponovo OLR, zadate da pronađe poruku po nekoj ključnoj reči i koji trenutak kasnije na ekranu vas već čeka poruka jednog od velikog broja vaših *Sezam* prijatelja, koji je problem već rešio.

Ovakav komfor u radu sa *Sezamom* je moguće sasvim jednostavno postići – sve što vam je potrebno je dobar komunikacioni program (*Procom*, *Telix* ili *Telemate*), SOR – *Sezam Off-Line Reader* i malo strpljenja da se čitav sistem postavi u jednu celinu. Ovde ćemo se pozabaviti samim SOR-om, pošto on predstavlja novost na *Sezamu*.

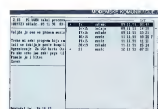
Off-Line Reader ideja je relativno nova stvar u komunikacijama. *Reader*-i su brzo stekli popularnost i skoro za svaki popularniji BBS sistem može se naći odgovarajući OLR – za najpopularniji PC Board i više različitih. *Sezam* je dosta kasno dobio svoj OLR, ali treba imati na umu da je u sam sistem od početka ugrađeno niz opcija koje obavljaju veliki deo posla koji inače OLR-i na drugim sistemima rade (pakovanje poruka i prenos, off line priprema poruka i slanje). SOR je delo autora ovog teksta, preuzima se sa samog *Sezama* i besplatan je za sve redovne korisnike *Sezama*. Prva verzija programa je nastala krajem jula meseca, da bi se, nakon odličnog prijema kod korisnika i niza njihovih sugestija, ubrzo pojavi-



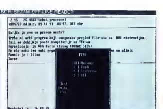
Prva poruka po ulasku u *SOR*



Pritiskom na taster "/" dobija se spisak konferencija



Pritiskom na taster [illegible] dobija se spisak poruka u izabranoj konferenciji



Pritiskom na taster ALT-F dobija se maska za unos kriterijuma za pretraživanje baze

la prva zvanična verzija (1.00) a nedavno, početkom oktobra, i najnovija verzija 2.30. Program je, zahvaljujući Miljanu Jovanoviću (mjova na *Sezamu*) opremljen i svom potrebnom dokumentacijom.

PREUZIMANJE SA SEZAMA...

Kada preuzmete SOR230.ZIP iz R:\COM direktorija i raspakujete arhivu, ostaje vam nešto posla oko instalacije. S obzirom da su uputstvo, help i ostale informacije na našem jeziku, paket je predviđen za automatsku konverziju prema kodnom rasporedu koji vi koristite. Dovoljno je da pokrenete STARTUJ, odgovorite na pitanje u kom od ispisanih redova vidite kompletan skup naših slova (čćđšž) i nakon toga sve datoteke će biti konvertovane (UYU program za konverziju i format datoteka su delo Dejana Ristanovića, takođe dostupno na *Sezamu*).

Nakon toga potrebno je da bilo kojim editorom koji radi sa čistim ASCII tekstom unesete potrebne informacije prema uputstvu iz datoteke SOR.DOC (potrebno je da unesete vaše korisničko ime, postavite nekoliko direktorija u koje program smešta ili prima podatke, i navedete nazive vašeg editora i komunikacionog programa – ostale vrednosti možete prihvatiti za početak kako su predložena). Sada možete startovati

SOR

Sačekaće vas prazan ekran sa porukom MESSAGE BASE EMPTY (baza poruka prazna). Poruke se ubacuju u bazu potpuno automatski – potrebno je samo da ih prenesete sa *Sezama*. Da biste to izveli, pozovite *Sezam* i odmah u osnovnom meniju izaberite opciju „1 AUTOMATSKI REŽIM". Na sva tri pitanja odgovorite sa „Da" i kada sistem javi da je spreman za prenos, u vašem komunikacionom programu pokrenite takozvanu *download* proceduru. Nakon završenog prenosa, pozdravite se sa *Sezamom* opcijom „9 Kraj" iz osnovnog menija.

Sada ponovo startujte SOR na sledeći način:

SOR /A

i program će sam raspakovati primljenu arhivu, preuzeti poruke iz nje i nakon toga na ekranu će se naći prva nova poruka koju još niste pročitali (slika 1). Na vrhu ekrana je standardno zaglavlje poruke koje *Sezam* ispisuje, u sredini je sam tekst poruke, a na dnu komandna i statusna linija. Kao i na samom *Sezamu*, i u SOR-u su poruke grupisane u konferencije (kao što su PC.PROG, PC.HARD, KOMUNIKACIJE itd), a u okviru svake konferencije i teme (kao što su mreže, štampači, diskovi itd. u konferenciji PC.HARD). U statusnoj liniji ćete pronaći naziv konferencije u kojoj se trenutno nalazite, redni broj teme u toj konferenciji i redni broj poruke u samom SOR-u, te ukupan broj poruka na tu temu. Ukoliko je uz poruku prikačena i datoteka, u desnom gornjem uglu će biti dat njen naziv, dužina i vreme koje je potrebno da se datoteka prenese na vaš računar.

...I KRETANJE PO SOR-U

Kada pročitate poruku, pritisnete taster „→" (ili kursor desno) i program vam daje sledeću poruku na istu temu. Ukoliko u tekućoj temi nema više novih poruka, program ispisuje odgovarajuću poruku i nudi da još jednim pritiskom na „→" pređete na poruke iz sledeće teme. Naravno, tasterom „←" (ili kursor levo) uvek se možete vratiti korak nazad. Ipak, glavne pogodnosti SOR-a počinju u trenutku kada odlučite da po bazi poruke pravite malo „krupnije korake". Pritisnete taster „/" (kosa crta) i na ekranu se pojavljuje spisak konferencija (slika 2). Dovedete kursor na onu koja vas zanima i nakon ENTER naći ćete se na prvoj novoj poruci u toj konferenciji. Ukoliko vas zanima neka određena tema, nakon „/" pritisnite isti taster još jednom i spisak konferencija će se „razviti" u detaljan spisak tema u svakoj od njih.

Posebnu pogodnost kod praćenja diskusije SOR nudi u takozvanom REFERENCE režimu. Naime, na *Sezamu* su poruke složene redom kojim stižu na sistem. Kako poruke najčešće predstavljaju odgovor na neku drugu poruku, čitav taj niz formira jedno, često veoma složeno stablo koje se teško prati po vremenskom redosledu. SOR REFERENCE režim omogućava da poruke čitate po prirodnom toku diskusije, bez obzira na vreme kada su pristigle – prvo čitate osnovnu poruku, zatim odgovor na nju, zatim odgovor na taj odgovor i tako sve do kraja liste, kada se vraćate na sledeći odgovor na osnovnu poruku (naravno ukoliko ga ima). Ukoliko vam se neki lanac diskusije učini nezanimljivim, pritisnite taster „." (tačka) i na ekranu dobijate listu poruka u tekućoj temi. Prva poruka u svakom od lanaca je obeležena znakom „+" (slika 3) – dovoljno je da pritisnete ovaj taster i program će vas odvesti na sledeći lanac diskusije.

Pored ovog „ručnog" kretanja kroz bazu, SOR omogućava i pretraživanje baze po nekoliko kriterijuma. Nakon tastera ALT F na ekranu se pojavljuje maska (slika 4) u kojoj možete kombinovati kriterijum pretraživanja – po vremenu poruke, autoru, nazivu prikačene datoteke i teksta koji treba pronaći u poruci. Sa istim parametrima radi i naredba FLAG, ali njen rezultat je marker koji se automatski dodeljuje svim porukama koje se uklope u zadati kriterijum. Kada su poruke na ovaj način obeležene (moguće je i ručno intervenisati na svakoj pojedinačnoj poruci), izabrani skup se može odštampati bilo na štampač bilo u datoteku, može se naredbama NF (*next flag*) i PF (*previous flag*) kretati od jedne do druge obeležene poruke i, konačno, mogu se naredbom REMOVE sve obeležene poruke izbaciti iz baze.

Ipak, i pored svih pogodnosti koje SOR nudi pri čitanju poruka, reklo bi se da je najkorisnija opcija direktno odgovaranje. Ukoliko želite da reagujete na neku poruku, držite pritisnut SHIFT taster i kursor „prevlačite" preko redova poruke za koje smatrate da zahtevaju komentar. Zatim će vas jedno ALT R (od *reply* – odgovor) odvesti pravo u vaš editor, u kome će se naći i prethodno obeležen citat. Nakon izlaska iz editora, poruka je zapamćena u obliku koji je spreman za slanje na *Sezam*. Na vaš odgovor se možete vraćati proizvoljan broj puta, bilo ponovnim odgovaranjem na istu poruku, bilo pritiskom na ENTER u listi vaših odgovora koja se dobija tasterom ALT V.

Ukoliko želite da prekinete rad sa programom, otkucajte Q (od *quit*) ili pritisnite ALT X. Sledeću put kada se vratite u SOR, sačekaće vas ista ona poruka na kojoj ste prethodno stali i, naravno, svi odgovori koje ste do tog trenutka napisali. Ipak, tipičnu seansu sa SOR-om završavate naredbom SEEN (ALT S), čime poruke proglašavate za stare. Potom, ukoliko imate odgovora koje želite da pošaljete, pritisnete taster F10 – SOR će tada od svih vaših poruka i naredbi (SOR omogućava i zadavanje standardnih *Sezam* naredbi putem prozora koji se dobija sa ALT C) formirati *Sezam* EXEC datoteku i startovati vaš komunikacioni program. Ukoliko ste i u njemu predvideli automatsku proceduru za slanje, vaši odgovori će biti poslati na *Sezam* bez ikakve intervencije.

Program je u stanju da izvrši nekih šezdesetak komandi, od kojih smo ovde spomenuli samo desetak. U svakom trenutku možete tasterom F1 dobiti ekran sa kratkim informacijama o načinu i nameni svake od komandi. Ukoliko primetite da određenu sekvencu naredbi često ponavljate ili vam se ne sviđa kombinacija tastera kojom se neka naredba ostvaruje, možete program prilagoditi svojim zahtevima zahvaljujući MACRO naredbama. Konačno, ako i dalje osećate da vam nešto nedostaje ili imate predloga za poboljšanja, na raspolaganju vam je posebna tema u konferenciji SEZAM (SEZAM.sor) u kojoj možete izneti svoje primedbe i prodiskutovati sa ostalim članovima i autorom.

NORTONOVE ALATKE

Početkom leta izašla je nova, šesta po redu verzija paketa *Norton Utilities*. Mnogi korisnici ovog popularnog paketa zaustavili su se na veriziji 4.5. Da li im nova verzija daje dovoljno razloga da promene svoje navike?

Norton Utilities spada u kategoriju DOS alatki, odnosno paketa koji proširuju mogućnosti operativnog sistema i na taj način olakšavaju svakodnevni rad na računaru. Paket ove vrste sadrže alate za obnavljanje obrisanih datoteka i direktorijuma, programa za ubrzavanje komunikacije sa diskom (disk cache), uklanjanje fragmentacije diska, navigaciju po stablu direktorijuma i drugo.

Testirana je verzija 6.01 iz avgusta 1991. Paket se u američkoj verziji distribuira u dve kompleta od po 4 diska formata 5.25" i 3.5". Dve diskete formata 5.25" su kapaciteta 360 K i dve kapaciteta 1.2 MB. Diskete formata 3.5" imaju kapacitet 720 K. Programe prati pet priručnika: od *Installation Guide* preko *Command Line Usage Guide*, *User's Guide*, *NDOS Reference Manual* do *Disk Explorer*-a. Oznake na disketama su *Installation*, *Utilities #1*, *Utilities #2* i *Emergency*.

NU 6 zahteva računar IBM PC, XT ili PS/2 klase, DOS verzije 2.0 ili novije (MS DOS 5.0 kompletno je podržan) i pamćenje 512 K memorije. Obezbeđena je podrška za miša, kao i za velike diskove pod DOS-om 4.01 i 5.0, Compaq DOS-om, Zenith DOS-om i DR DOS-om 5.0. Za punu instalaciju programa potrebno je oko 2.8 MB slobodnog prostora na disku. Instalirani programi se na disku nalaze u komprimovanom obliku radi uštede na prostoru. Na sporijim računarima (sa procesorom 8088, 8086 ili 80286 na 10 ili 12MHz) vredi raspakivati programe.

BRZ POGLED NA PROGRAME

Programi u paketu podeljeni su u nekoliko grupa: alati za obnavljanje obrisanih datoteka i intervencije na disku, defragmentaciju diska i ubrzanje pristupa, zaštitu podataka, alate opšte namene i uslužne programe koji se pokreću iz komandne linije.

Norton Utilities 6.0 podržava DOS 5.0, preklapanje procesa (*task switching*), evidenciju obrisanih datoteka, mogućnost oporavka od slučajnog formatiranja diskova i naredbu LOADHI za izvršavanje programa iz zone visoke memorije.

Svi programi koji se izvršavaju preko menija imaju kontekstno osetljiv help koji se aktivira pritiskom na taster F1. Pored toga, svi programi u paketu imaju help koji se iz komandne linije dobija uvođenjem argumenta /?. Ukoliko se program instalira na računaru čiji je operativni sistem DOS 5.0, INSTALL će informacije o korišćenju programa dodati i u odgovarajuću datoteku operativnog sistema.

NORTON

Lak pristup svim programima omogućava vam program Norton. Tu su programi predstavljeni u jedinstvenom meniju sa kratkim opisima funkcije svakog od njih i načinom upotrebe. Interesantno je da se meniji mogu menjati: u njih možete dodati programe ili ukloniti one koje ne koristite. Taj posao će vam oduzeti minut vremena: iz MENU menija izaberite ADD TOPIC. Neka nova tema menija bude npr. KOMUNIKACIJE; unesite ime nove teme, odnosno stavke menija i njen kratak opis. Zatim ćete ponovo iz ADD menija dodati komandu SEZAM koja će pozivati vaš komunikacioni program sa zadatkom da uspostavi vezu sa Sezam BBS-om. Ograničenja postoje, ali ćete teško doći u situaciju da ih srećete: rad se o tome da tema može biti najviše desetak, a čak su već iskoristili programeri „Symantec"-a.

Norton, glavni program u paketu, toliko je fleksibilan da vam dozvoljava da dodajete, brišete i menjate bilo koji element već postojećih stavki u menijima. Možete na primer izbrisati sve komande i zameniti ih sa komandama po vašem izboru. Možete izbrisati sve teme i zameniti ih sa najviše deset vlastitih tema.

Postoji, naravno, i „zona sumraka": ako ste program toliko izmenili da ne znate kako da ga vratite u prvobitno stanje, moraćete da ponovo pokrenete *Install* program sa originalnih disketa. U *Install* programu postoji i opcija kojom se onemogućava menjanje menija *Norton* programa. Ako program instalirate za druge i ako su vaši korisnici baš uporni, možete podesiti atribute datoteka NORTON.INI tako da dozvoljavaju samo čitanje datoteke (read-only).

Zoran i Marijan Kehler

OBNAVLJANJE IZGUBLJENIH PODATAKA

Programi iz ove grupe deo su kompleta programa obaveznih za svakog korisnika. *Norton Utilities* je pradeda svih softverskih paketa ove vrste. Verzija kakvu danas imamo je od malog skupa programa za nalaženje obrisanih datoteka i restauriranje tabela particija napredovala do nečega što asocira na švajcarski vojnički nož. Tu su programi za defragmentaciju datoteka, zaštitu od slučajnog brisanja, oporavak od nenamernog formatiranja, kontrolu stanja diska, za preventivno održavanje diska i direktno editovanje sadržaja diska i spašavanje podataka.

Program koji ćete sigurno koristiti jeste *Norton Disk Doctor* (NDD). On vam omogućava automatsko dijagnosticiranje i popravku oštećenog sadržaja diska. Da bi vam uštedeo trud, NDD će sam proveriti tabelu particije, *boot* zapis na disku, FAT tabele i strukturu datoteka i direktorijuma. Takođe, uradiće i posao sličan onome koji obavlja CHKDSK komanda DOS-a, ali na mnogo temeljniji i sigurniji način. Testiranje će potrajati duže ako se odlučite za test površine diska, koji je preporučljivo izvršiti bar jednom mesečno, naročito na starijim diskovima.

NDD se može izvršavati na mreži, ali se njime ne mogu testirati mrežni diskovi. Nije moguće testirati diskove sa brojem cilindara većim od 1024. Program poznaje sve načine particionisanja diskova: od DOS-a 3.31 preko DOS-a 4.0, 5.0, *SpeedStor*-a i *Disk Manager*-a. Treba obratiti pažnju na to da *Norton Disk Doctor* služi prvenstveno za otkrivanje nepravilnosti na disku – dakle više za postavljanje dijagnoze, a manje za lečenje.

Disk Editor vam omogućava da pregledate i editujete celokupan sadržaj tvrdih i flopi diskova. Možete raditi čak i na diskovima koje DOS ne prepoznaje. Objekti koje vidite iz programa su tabela particija, *boot* zapis, FAT tabele i bilo koja grupa sektora ili klastera na disku.

Postoje situacije kada samo *Disk Editor* može oporaviti disk ili spasiti podatke: kada NDD ne može da ispravi greške u strukturi zapisa na disku, kada podaci neočekivano nestaju ili se pojavljuju na disku, kada treba izvući podatke sa neispravnog diska, menjati podatke u datotekama nezavisno od aplikacionih programa, kada je uništen sadržaj direktorijuma i kada želite da imate uvid u sadržaj diska formatiranog pod operativnim sistemom različitim od DOS-a.

Ako *Disk Editor* želite da iskoristite kao prvu pomoć u rešavanju konkretnog problema, nemojte ga instalirati na disk ako se na njemu već ne nalazi. Postoji, naime, mogućnost da ima obrišete deo podataka koji želite da spasete.

Disk Editor je izuzetno moćan program. Naše iskustvo u radu sa njim izuzetno je povoljno. Naravno, nismo za potrebe testiranja programa namerno oštetili podatke na disku, niti brisali *boot* zapis ili tabelu particija. Prema Marfijevim zakonima, problemi te vrste nastupaju neočekivano, ali se prepoznatljivom zakonitošću. Najopasnije je podleći panici i koristiti se moćnim alatima nasumice. NDD nije dovoljan za rešavanje problema, ne sme se ni slučajno povarovati da je nedostatak samo u lošoj FAT tabeli ili pogrbljenim klasterima. Ako NDD otkrije nedostatke, odmah treba pokrenuti *Disk Editor* i slediti uputstva za krizne situacije. U kompletu uputstava koji prate paket nalazi se odličan *Disk Explorer* priručnik. On obiluje informacija o načinu rada diskova, formatima podataka, logičkim strukturama koje koristi operativni sistem i mogućim greškama. Procedure za otklanjanje kvarova podeljene su u četiri grupe: spašavanje podataka, popravak direktorijuma, popravak sistemskih područja, popravke na nivou editovanja apsolutnih sektora na disku. Za rešavanje problema iz ove četiri oblasti stručnjaci Symantec-a daju detaljna precizna uputstva za rešavanje.

Ostali programi iz ove grupe uglavnom su standardni proizvodi kakvi se sreću i u paketima drugih proizvođača. Program *Disk Tool* donosi interesantnu ideju: čuvanje sadržaja CMOS memorije, *boot* zapisa i tabele particija vašeg računara. Kao i *Disk Editor*, ova opcija pomogla nam je više nego jednom u oporavku umrtvljenog računara. *Erase Protect* dobro će doći korisnicima koji se olako odlučuju na brisanje datoteka: on obrisane datoteke premešta u drugo manje korišćeno područje diska, da ne bi bile prepisane. Program je rezidentan i radi na mrežnim diskovima. *Image* čuva sliku sistemskih područja diska u datoteci IMAGE.DAT.

Unerase vraća obrisane datoteke. Program podržava *Mirror* i *Delete Tracking* (evidencija o obrisanim datotekama) iz DOS-a 5.0 tako što pri izvršavanju proverava postojanje *Mirror* i *Image* podataka da bi oporavak izvršio na osnovu njihovih informacija.

UnFormat će oporaviti disk koji je nepažnjom ili slučajno formatiran, čiji je sadržaj oštećen dejstvom virusa ili zbog nestanka električne energije. Program oporavlja i sadržaj diskete formatirane pomoću *Safe Format* programa. Oporavak će biti pokušan prvo na osnovu sadržaja IMAGE.DAT datoteke, ako ona postoji, i samo na osnovu sačuvanih podataka.

UBRZAVANJE RADA RAČUNARA

Programi za ubrzavanje rada diska, defragmentaciju datoteka, podešavanje *interleave* faktora diska uvek su bila jaka strana *Norton* paketa. S pravom su to i ovde: ipak nam je *Symantec* pripremio pouzdane alatke, dosta unapređene u odnosu na ranije verzije.

Speed Disk je, uz analogan program iz paketa *PC Tools*, verovatno najviše korišćeni alat za defragmentaciju diska. On reorganizuje fizički raspored datoteka i direktorijuma na disku da bi se minimiziralo kretanje upisno-čitajuće glave. U osnovi, on obavlja dva zadatka: formira zonu slobodnog prostora na disku na kraju diska (to jest na unutrašnjim cilindrima, do kojih glava najduža putuje) i uklanja fragmentaciju datoteka. Program poštuje razne šeme za zaštitu tajnosti podataka i zaštite od kopiranja programa tako što skrivena datoteke i direktorijume ostavlja na njihovim ranijim mestima. Nije moguće njegovo korišćenje na mrežnim diskovima.

Dosadašnje verzije bile su neoprostivo spore čak i na brzim mašinama. Poslovično je bilo da pokretanje *Speed Disk*-a znači obavezno posmatranje crtanog filma u kome program prikazuje kako premešta blokove po disku. Toga, na sreću, više nema: ceo posao obavlja se izuzetno brzo. Važna je i pouzdanost koja je dovedena do tog nivoa da programu ne smeta da se prekine usred rada – da podaci i tada ostaju neoštećeni. Naravno, nemojte pokušavati da isključite napajanje računara dok se ovaj program izvršava – rezultati će verovatno zahtevati intervenciju sa *Disk Editor*-om. Iako u uputstvu stoji da je moguće da se *Speed Disk* ne slaže sa pojedinim rezidentnim programima koji pristupaju disku, praktično iskustvo (možda nismo dosta srećni) nam je pokazalo da ozbiljnih problema sa te strane nema.

Calibrate je odgovor *Symantec*-a na slične alate drugih proizvođača. Zbog ranijeg iskustva sa programom *Spin Rite II* bili smo radoznali da vidimo kakvi će biti rezultati primene iskustva Petera Nortona na istom poslu. Program ove vrste trebalo bi da optimizira brzinu i pouzdanost upisa i čitanja podataka na disk. Da bi optimizirao brzinu, *Calibrate* proverava faktor preplitanja diska, a zatim radi nedestruktivno *low level* formatiranje površine diska, podešavajući sada podatke na optimalni faktor preplitanja ili na onaj koji ste sami izabrali.

Symantec tvrdi da *Calibrate* ne oštećuje podatke koji se nalaze na disku i da im čak ni prekid napajanja ne može nauditi. Ovu drugu tvrdnju nismo pokušavali da proverimo prva među nama, ali je program potpuno korektno obavio posao na nekoliko IDE diskova, ali je obično faktor preplitanja ostajao nepromenjen. Isproban je i na jednom starom MFM disku, paralelno sa *Spin Rite II* programom. Oba programa postigla su gotovo iste rezultate, ali je *Calibrate* bio nešto brži. Naravno, i ovde treba voditi računa da program može biti osetljiv na prisustvo rezidentnih programa.

Norton Cache je disk keš program koji bi prema opštoj teoriji trebalo da ubrza rad sa diskom. „Symantec"-ov keš program podržava i keširanje upisa na disk. I ovde se uputstvo ograđuje od eventualnih grešaka: korisniku se preporučuje da keš skine pre izvršavanja programa za defragmentaciju diska ili alata kao što je *Disk Doctor*. Programi u paketu strogo se drže toga, pa će *Disk Doctor*, *Speed Disk* i *Calibrate* automatski skinuti *Norton Cache*.

Program je moguće instalirati u CONFIG.SYS datoteci ili iz komandne linije. Najjednostavnije je prihvatiti podrazumevane vrednosti – keš instalirati samo sa NCACHE /INSTALL. *Norton Cache* će prepoznati da li postoji XMS ili EMS memorija, kao i veličinu te memorije, i konfigurisati se prema tome. Nemojte koristiti FASTOPEN program iz DOS-a, koji posreduje u pristupanju FAT tabelama, jer može prouzrokovati probleme u radu sa NCACHE programom.

U program je ugrađen *IntelliWrites* sistem za pisanje na disk u pozadini, za vreme izvršavanja drugih programa. Ovo se može i isključiti ako postoji opasnost od konflikata sa raznim disk drajverima.

Norton Cache uporedili smo sa najčešće korišćenim keš programom SMARTDRV-om iz DOS-a i *Windows* paketa, koji korisnici obično i ne primećuju u radu. Symantec-ov program primetno je brži, a poseduje i korisnu mogućnost pisanja na disk u pozadini, za vreme izvršavanja drugih programa. Pokazalo se, međutim, da korišćenju ove opcije treba prići oprezno. Na računaru klase 386/33MHz sa „Quantum" SCSI diskom, „Adaptec" SCSI kontrolerom i „Adaptec" ASPI4DOS disk drajverom došlo je do zaglavljivanja programa. Korišćen je *Windows 3.0* a, da prethodno suprotno navodima u uputstvu keš programa, nije u SYSTEM.INI dodata linija koja omogućava da zajedno sa *Windows* programom radi i program koji piše na disk u pozadini. No nakon dodavanja linije „VirtualHDIrq=false" u „[386Enh]" sekciju inicijalizacione datoteke sve je funkcionisalo ispravno. Ovakav problem nije primećen na računarima sa IDE disk sistemima.

Posle početnih problema na računaru koji je korišćen za testiranje, *Norton Cache* pokazao se pouzdanim i korisnim. Svoje vrline pokazuje naročito u radu sa bazama podataka (velike datoteke, čest pristup), ali i u radu na tipično programerskim zadacima: prevođenju i povezivanju programa. To se pokazalo u korišćenju RTLink linkera, koji je specifičan po tome što intenzivno komunicira sa diskom dok kreira privremene datoteke u procesu linkovanja. Sve u svemu, „Symantec" je napravio dovoljno dobar keš program, da ga možete prihvatiti umesto vašeg standardnog SMARTDRV programa. Pitanje je, međutim, da li bi izdržao strog uporedni test sa rasnim keš programima iz klase *PC Kwik, HyperDisk* i drugim.

ZAŠTITA PODATAKA

Preventivna zaštita podataka drugi je način borbe protiv problema koji nastaju zbog nepažnje, neznanja, ali i smišljenih pokušaja da se podaci pregledaju ili izbrišu. Načini koje omogućavaju programi u *Norton Utilities* su podeljeni u dve grupe: zaštita od pisanja po kritičnim zonama diska i zaštita od neovlašćenog pristupa podacima.

Disk Monitor spada u prvu grupu programa. Pomoću njega možete na ekranu dobiti indikator rada diska („Disk Light"), zaštititi disk od pisanja („Disk Protect") i parkirati glave diska pre isključivanja računara („Disk Park"). Ova treća opcija je od manjeg praktičnog značaja u vreme kada se već nom koriste IDE, ESDI i SCSI diskovi kod kojih je parkiranje glava automatsko.

Diskreet je program za zaštitu od neovlašćenog pristupa podacima i računaru. Važne datoteke se kodiraju i pristup je moguć samo uz lozinku. Takođe, moguće je na disku odvojiti prostor za NDisk, objekat koji ima osobine skrivene datoteke, a ponaša se kao poseban tvrdi disk, zaštićen lozinkom.

Program koristi dva načina kodiranja. Prvi je DES (Data Encription Standard), usvojen od strane američke vlade kao bezbedan način kodiranja informacija, a drugi je delo „Symantec"-ovih programera. Razlika koju korisnik vidi na prvi pogled, ako nije zainteresovan za razbijanje šifre, jeste u brzini – DES postupak je dosta sporiji.

Napomenimo da je testirana verzija *Norton Utilities* namenjena američkom tržištu. Verzije kupljene u Evropi obično nemaju ugrađen DES algoritam zaštite, jer se on nalazi na listi proizvoda visoke tehnologije čiji je izvoz ograničen sigurnosnim razlozima.

Program *Wipe Information* je naslednik *WipeDisk* i *WipeInfo* programa iz ranijih verzija paketa. On fizički briše izabrana područja diska, upisivanjem nekog niza bajtova na mesta gde su se nalazile datoteke, ili na lokacijama koje korisnik bira. Ovo ispravlja nedostatak DOS-a, koji čak ni kod formatiranja ne uklanja stari sadržaj diska, već ga samo čini logički nevidljivim. Problem je izgleda toliko aktuelan da je američko ministarstvo odbrane donelo preporuku o načinu brisanja poverljivih informacija sa diska. Program nije nimalo naivan i sa njime se mora postupati oprezno. Nakon delovanja *Wipe Information* čak ni *Disk Editor* nije uspevao da nađe ostatke podataka na disku.

ALATI OPŠTE NAMENE

Norton Utilities je tradicionalno dobro opremljen alatima opšte namene, koji se verovatno najviše i koriste. Vrlo je bitno da uvek postoji i mogućnost korišćenja iz komandne linije, pored rada sa menijima. Neki programi zaslužuju više pažnje, a neke je dovoljno samo pomenuti – vi ćete se korišćenjem uveriti u njihovu pouzdanost i kvalitet.

Navigacija po direktorijima: *Norton Change Directory*, NCD

Verovatno je svaki iskusniji korisnik PC računara koristio *Norton Change Directory* (NCD) program, možda i ne znajući da je reč o delu *Norton Utilities* paketa. Izgleda da je nekako ovaj program prešao u opšte vlasništvo čak i kod onih koji se iskreno zalažu za poštovanje autorskih prava na programe.

NCD se može koristiti iz komandne linije i interaktivno pomoću grafičkog prikaza stabla direktorijuma. U komandnoj liniji nije potrebno navoditi puna imena odredišta, jer će program već pomoću nekoliko slova pokušati da nađe direktorijum koji želite. Norton CD po prvom aktiviranju pretražuje stablo direktorijuma da bi ga sačuvao u datoteci TREEINFO.NCD. To omogućava da kasniji prolaz stablom budu brži.

Programu je dodata i mogućnost editovanja stabla. Opcije su „Copy Tree", kada se taj deo stabla ne briše sa polaznog mesta, i prava „Prune and Graft", kada se deo stabla stvarno premešta na drugu poziciju.

Directory Sort: sortiranje direktorijuma

DS služi za sortiranje direktorijuma po imenu, tipu, datumu i vremenu kreiranja datoteka i direktorijuma. Moguć je rad sa menijima i rad iz komandne linije. Redosled datoteka i direktorijuma može se u interaktivnom radu menjati.

System Information

Norton System Information je za drže provere računara i merenja performansi procesora i diska. Sa te strane, čak iako ne predstavlja nikakav standardni skup procedura za testiranje, SI pruža uvid u relativni odnos performansi različitih sistema.

Parametri koje prepoznaje *System Information* obuhvataju informacije o instaliranoj memoriji, hardverskoj konfiguraciji sistema, hardverskim i softverskim prekidima, iskorišćenju memorije, parametrima tvrdih i flopi diskova, instaliranom mrežnom softveru. SI vam može pomoći kod instaliranja dodataka u sistem, pregledom instaliranih portova, spiskom iskorišćenih interapta i adresa I/O uređaja.

Pronalaženje datoteka: *File Find* i *File Locate*

Programi *File Find* i *File Locate* nalaze datoteke za koje smo zaboravili gde se nalaze na svim diskovima, uključujući i mrežne. FF je postao deo radne okoline gotovo svakog korisnika. FL je program koji se pokreće iz komandne linije i poznaje manje opcija od svog starijeg brata.

File Find može tražiti datoteke prema njihovom nazivu ili prema ključnoj reči koju sadrže. Na pronađenim datotekama može da obavi razne operacije: postavljanje atributa, vremena kreiranja ili da njihove nazive ispiše u batch datoteku. Ako se izabere ova poslednja opcija, FF vam omogućava da zadate komandu koja će se izvršiti nad svim pronađenim datotekama.

FF pretražuje i logičke i fizičke mrežne diskove. Može se zadati opcija da datoteke traži prema zadatom vlasniku.

Podešavanje radne okoline: *Norton Control Center*

Norton Control Center kontroliše hardverske parametre vaše radne okoline. Možete ih lako menjati, bez intervencije u sistemskim datotekama. Podešava se oblik kursora, boje ekrana, paleta boja, bira se video mod, brzina ponavljanja tastera, brzina miša, konfigurišu serijske portove, na raspolaganju su celi brojača vremena, razni formati prikaza vremena, datuma i brojeva u zavisnosti od izabrane zemlje.

NCC se koristi interaktivno ili iz komandne linije, što omogućava da se upotrebi u AUTOEXEC.BAT proceduri i računar podesi odmah po uključivanju. Parametri podešeni u interaktivnom radu mogu se sačuvati u posebnoj datoteci i želim iskoristiti iz komandne linije naredbom „NCC ime_datoteke /SET".

Poboljšanje *batch* obrade: *Batch Enhancer*, BE

BE obogaćuje *batch* obradu kontrolom nad ulazom podataka, grafičkim dodacima, zvukom i mogućnošću obrade u unapred zadato vreme. Primena se lako nalazi: *batch* procedure sa menijima za aktiviranje aplikacija, omogućavanje izbora rezidentnih programa u AUTOEXEC.BAT datoteci, jednostavne animirane demonstracije, reklame.

Iako se *Batch Enhancer* može koristiti u običnim DOS *batch* programima, njegova prava snaga leži u BE skript datotekama. Komande ostaju iste, ali ih BE izvršava direktno, bez posredovanja DOS-a, pa je brzina izvršavanja primetno veća. BE skript programi pokreću se sa BE ime_skripta. Izvestan broj naredbi *Batch Enhancera* postoji u NDOS-u, zameni za COMMAND.COM. Primer menija kreiranog pomoću BE programa dat je na listingu 1.

BE ASK očekuje ulaz sa tastature. Naredba setuje vrednost ERRORLEVEL promenljive, na osnovu čega se može granati izvršavanje procedura. Može se zadati ista tastera koji se prihvataju kao i vrednost koja se pretpostavlja ako je odgovor nepravilan ili je isteklo vreme predviđeno za čekanje.

BE REBOOT će izvršiti softversko resetovanje računara, uz kontrolno pitanje da li to stvarno želite. Korisno ako u *batch* proceduri menjate konfiguraciju sistema.

BE SA kontroliše boje i ostale karakteristike prikaza na ekranu. Zahteva da je na računaru instaliran ANSI.SYS drajver. Mogu se podesiti boja teksta, pozadine i ruba ekrana, inverzni, podvučen i trepćući atributi teksta ili obrisati ekran.

Ostale alatke

Još nekoliko programskih alatki kompletira sadržaj *Norton Utilities*. *File Attributes* služi za prikaz i podešavanje atributa datoteka i direktorijuma. *File Date* menja vreme i datum kreiranja datoteka. Veličinu datoteka naći ćete sa *File Size*. FS prikazuje veličinu datoteka, kao i prostor koji je za njih odvojen na disku. Može poslužiti za određivanje koliko je prostora na odredišnom disku potrebno za kopiranje datoteka. *Line Print* formatira datoteke i šalje ih na štampanje. Možete podesiti margine, broj linija na strani, način spisivanja zaglavlja ili čak štampati na Post Script štampačima. *Text Search* traži zadati tekst u datotekama, na celom disku ili specijalno u prostoru gde se nalaze obrisane datoteka. Umesto FORMAT naredbe iz DOS-a, *Norton Utilities* vam nudi da koristite *Safe Format*, koji je brži, ostavlja mogućnost obnavljanja sadržaja slučajno formatirane diskete.

ČVRSTO NA VRHU

Symantec je licencirao deo NDOS-a od kompanije „JP Software Inc.", tvorca 4DOS komandnog procesora. 4DOS je već od ranije poznat kao koristan i pouzdan proizvod. Uzimanjem programa pod svoje okrilje, Symantec ga je obogatio novim naredbama, učinio kompatibilnim sa novom verzijom DOS-a, *Windows* programom i mrežnim operativnim sistemima. NDOS dolazi sa kompletnim priručnikom i lepo se uklapa u ostatak *Norton Utilities*. Pošto se radi o izuzetno značajnom proizvodu, o njemu pripremamo poseban članak za sledeći broj.

Norton Utilities čvrsto drži svoje mesto u klasi paketa koji bi trebalo da zadovolje veoma veliki broj potreba korisnika. Konkurencija nije mala: *PC Tools*, *Mace Utilities* i drugi bore se za deo tržišta koji pripada Symantec-u. Za razliku od konkurentnih proizvoda, *Norton Utilities* nije se utopio u sopstvenoj veličini: programi su još uvek laki za korišćenje, opcije se intuitivno otkrivaju, programi ne brinju samo nicijativno po raflje instaliranom softveru. Interaktivni karakter alata ne čini ih prekompilkovanim za upotrebu. Ako ne volite šetnju kroz menije, za vas je kao stvoren poziv istih programa iz komandne linije.

Paket kao što je *Norton Utilities* vredeo bi mnogo manje bez kompleta priručnika. Uputstva za programe detaljno vode korisnika kroz rad, a tekstove o obnavljanju oštećenih podataka i oživljavanju diskova kao da je pisao sam Peter Norton. Posebno je *Disk Explorer*, priručnik za korišćenje *Disk Editora*, bogat izvor podataka za rad sa diskom, zbir iskustava u otklanjanju problema.

Za ponos paketa slobodno se mogu smatrati alati za obnavljanje i zaštitu podataka na disku, kao i programi za ubrzavanje rada računara. Izuzetak je *Norton Cache* program, koji je pokazao izvesnu ćudljivost u radu na opremi korišćenoj za testiranje, ali je besprekorno radio na nekim drugim konfiguracijama.

Posebne pohvale za NDOS komandni procesor. Bez ikakvog posebnog napora zamenjen je procesor na koji ukazuje SHELL= u CONFIG.SYS datoteci i COMSPEC= u AUTOEXEC.BAT, nikakvi posebni parametri nisu dodati i sistem je odmah proradio sa NDOS-om u visokoj memoriji podešen da *swapping* izvršava sa XMS memorijom. Iako je komandni procesor „SYMANTEC" nasledio od „JP Software", poboljšanja su očigledna.

Dugo ćemo još čekati da se pojavi programski paket koji će rešavati sve probleme svim korisnicima – posla za još jedan alat uvek ima. U međuvremenu, s pouzdanjem se možete odlučiti za kupovinu „Norton Utilities".

ANALIZA TEKSTA

U ovom članku će biti reči o „leksičkoj analizi". Biće izložene tehnike korišćene u izradi programskog modula koji ovu funkciju obavlja – „leksičkog analizatora" ili „skanera", kao i prikaz programske alatke koja ove module automatski generiše. Kao što ćemo videti, tema ovog teksta nije interesantna samo za nekog ko se bavi isključivo dizajnom kompajlera. Poznavanje ovih tehnika, a naročito upotrebe pomenute programske alatke, predstavlja veliku pomoć pri programiranju.

Igor Milosavljević

Šta je leksička analiza? Najbolje je početi primerom. Posmatrajmo naredbu dodele u jeziku paskal

res := val1 + 5

LEKSIČKA ANALIZA

Sa stanovišta sintakse i algoritma za sintaksnu analizu bilo bi neekonomično analizirati naredbu u ovakvom obliku. Postoji suviše nebitnih detalja – na primer konkretna imena promenljivih, broj blanko i tab znakova i eventualni komentar. Šta više, za sintaksnu analizu je nebitno da li se u naredbi dodele koristi „:=" kao u paskalu ili „=" kao u C-u ili u fortranu. Sintaksnom analizatoru (parseru), znači, treba proslediti do izvesne mere već „sažvakan" tekst izvornog koda. Na primer, navedena naredba bi se mogla transformisati u

„id" „:=" „id" „+" „num"

Da ne bude zabuna, ovo nije prosta zamena jednog teksta drugim – radi se o tome da je ulazni tekst „tokenizovan". Simboli „id", „:=", „+" i „num" se zovu „tokeni" i u implementaciji su to najčešće neke celobrojne vrednosti kojima su ovde, radi preglednosti, dodeljene simboličke oznake. Znači uloga leksičkog analizatora kao prve faze prevođenja je čitanje teksta izvornog koda i prosleđivanje niza tokena na svoj izlaz. Ovo je vizuelno prikazano na slici 1.

Token je najmanji element kojim se operiše u sintaksnoj analizi. U programskim jezicima se obično javljaju sledeći tokeni: brojne konstante, identifikator (ime promenljive), ključna reč, operator, literal itd. Način saradnje leksičkog analizatora i sintaksnog analizatora – parsera je prikazan na slici 2. Na zahtev parsera leksički analizator nastavlja se čitanjem teksta izvornog koda tamo gde je stao prilikom prethodnog poziva, i vraća sledeći token koji prepozna. Ova dva modula, uzgred rečeno, predstavljaju tzv. „front-end" kompajlera. Njihova implementacija ne zavisi od ciljne mašine čiji mašinski kod kompajler treba da generiše.

Osim tokenizacije izvornog koda, leksički analizator obavlja i druge funkcije. Jedna od najvažnijih je formiranje tabele simbola. Za svaki identifikator (ime promenljive) na koji naiđe, leksički analizator proverava da li je već registrovan u tabeli simbola. Ako nije, formira novi element tabele simbola i popunjava ga odgovarajućim informacijama. U ovoj fazi kompilacije su te informacije oskudne: niz znakova koji predstavlja samo ime promenljive i eventualno, u dijagnostičke svrhe, linija izvornog koda u kojoj je pronađena. Tabela simbola predstavlja ključnu strukturu podataka kojoj se obraćaju sve kasnije faze kompilacije, i namena joj je preslikavanje simboličkih imena u adrese memorijskih lokacija.

Postoji skup stringova za koje će, kada budu pronađeni u ulaznom tekstu, biti vraćen isti token. U gornjem primeru, za sve stringove oblika „slovo praćeno nizom slova i cifara" biće vraćen token „id". Reči u navodnicima iz prethodne rečenice predstavljaju „pravilo" pomoću kojeg se neki string identifikuje kao odgovarajući token. Ovo pravilo se na engleskom zove „pattern", što bi se moglo, dosta nezgrapno, prevesti kao „model" ili „shema".

Znači, opštija definicija leksičkog analizatora je: skaniranje ulaznog teksta u cilju prepoznavanja stringova koji zadovoljavaju određene osobine, i preduzimanje odgovarajućih akcija po tom prepoznavanju. U procesu kompilacije akcije su prvenstveno vraćanje tokena pozivajućem modulu, ukoliko je leksički analizator izveden kao potprogram. Akcije mogu da budu i izostavljene, što ima za posledicu odbacivanje pronađenog tokena. Ovo se radi sa nizovima znaka beline (whitespace) ili komentarima u programskom kodu. Leksički analizator, kao što ćemo videti, ima smisla realizovati i kao samostalan program. U tom slučaju, akcije vrše razne manipulacije sa pronađenim stringovima i skaner na izlaz ispisuje transformisan ulazni tekst.

DEFINISANJE MODELA

Da bismo napisali leksički analizator, potrebno ga je na neki način precizno definisati. Jedan od načina je zadavanje niza „pravila" sa uređenim parovima sledećeg oblika

<**model,** **akcija**>

Kao što smo spomenuli, „model" je opis svih stringova u ulaznom tekstu koji će biti prepoznati kao odgovarajući tokeni. Specifikacija jednog jednostavnog (i prilično nekorisnog) leksičkog analizatora je prikazana na slici 3.

Modele ćemo neformalno opisati na sledeći način:

identifikator	–>	**slovo praćeno nizom slova i cifara**
ceo broj	–>	**niz cifara**
belina	–>	**niz znakova SPACE, TAB i NL**
sve_ostalo	–>	**niz znakova koji ne spada ni pod jedno od navedenih pravila**

Ako je ulazni tekst sledeći paskal program

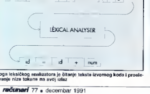
```
program testlex;
var
   i: integer;
begin
   i := 13
end.
```

naš leksički analizator bi trebalo da ispiše sledeći niz poruka

id: program	**other:**	**other: :=**
id: testlex	**id: integer**	**other: –**
other: ;	**other: ;**	**num: 13**
id: var	**id: begin**	**id: end**
id: i	**id: i**	**other: .**

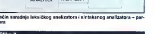
Sada ćemo se pozabaviti definisanjem modela. Neformalno definisanje koje smo koristili ima samo didaktički značaj. Da bismo mogli efikasno da programiramo, mora postojati sredstvo za formalizovano i koncizno definisanje modela. Takvo sredstvo su „regularni izra-

Slika 1



Uloga leksičkog analizatora je čitanje teksta izvornog koda i prosleđivanje niza tokena na svoj izlaz

Slika 2



Način saradnje leksičkog analizatora i sintaksnog analizatora – parsera

Slika 3

```
<identifikator, print "id:", pronađeni niz znakova>
<ceo_broj, print "num:", pronađeni niz znakova>
<belina, ne radi ništa>
<sve_ostalo, print "other:", pronađeni niz znakova>
```

Specifikacija jednog jednostavnog (i prilično nekorisnog) leksičkog analizatora

Listing 1 LEXAN.C

```
#include <stdio.h>
#include <ctype.h>

#define EOS '\0'
#define RESTART() i = 0; state = 0

/* u 'yytext' se po prepoznavanju
   tokena nalazi prepoznati tekst */
char yytext[256];

/* leksicki analizator u obliku funkcije
   koja se zavrsava rad dok se procita ceo
   ulazni tekst */
void yylex(void)
{
     int i = 0;
     int c;
     int state = 0;

     do
     {
          c = getchar();
          switch (state)
          {
          case 0:
               if (isalpha(c))
               {
                    state = 1;
                    /* formiranje 'yytext' */
                    yytext[i++] = c;
               }
               else if (isdigit(c))
               {
                    state = 2;
                    yytext[i++] = c;
               }
               else if (c == ' ' || c == '\t' || c == '\n')
                    /* nema potrebe za formiranjem 'yytext'
                       za za ovaj token */
                    state = 3;
               else if (c != EOF)
                    printf("other: %o\n", c);
               break;

          case 1:
               if (isalpha(c) || isdigit(c))
                    yytext[i++] = c;
               else
               {
                    yytext[i] = EOS;
                    ungetc(c, stdin);
                    RESTART();
                    /*** AKCIJA ***/
                    printf("id: %s\n", yytext);
               }
               break;

          case 2
               if (isdigit(c))
                    yytext[i++] = c;
               else
               {
                    yytext[i] = EOS;
                    ungetc(c, stdin);
                    RESTART();
                    /*** AKCIJA ***/
                    printf("num: %s\n", yytext);
               }
               break;

          case 3:
               if (c == ' ' || c == '\t' || c == '\n')
                    ;
               else
               {
                    ungetc(c, stdin);
                    RESTART();
                    /*** AKCIJA ***/
               }
               break;
          }
     } while (c != EOF);
}

/* glavni program, koji samo poziva 'yylex' */
main()
{
     yylex();
}
```

Implementacija skanera specificiranog na slici 3, na osnovu grafa sa slike 5

Listing 2 LEX.L

```
CIFRA       [0-9]
SLOVO       [A-Za-z]
%%
{SLOVO}({SLOVO}|{CIFRA})*       printf("id  %s\n", yytext);
{CIFRA}+                        printf("num  %s\n", yytext);
[ \t\n]+
.|\n                            printf("other  %s\n", yytext);
```

lex **program za skaner specificiran na slici 3**

Listing 3 REFORMAT.L

```
     /* prototipovi funkcija pozivanih iz akcija */
     char *underlined( char * );
     char *bold( char * );
BS \x08
%%

(_{BS}.)+             printf( "%s", underlined( yytext ) );
(.{BS}.{BS}.{BS}.)+   printf( "%s", bold( yytext ) );

%%
#define ESC          "\x1b"
#define UNDER_ON     ESC "-1"
#define UNDER_OFF    ESC "-0"
#define BOLD_ON      ESC "E"
#define BOLD_OFF     ESC "F"
#define BACKSPACE    8
#include <string.h>

char buffer[256];

char *underlined( char *text )
{
     int i, j;

     strcpy( buffer, UNDER_ON );
     j = strlen( buffer );
     for ( i = 2; i < yyleng; i += 3 )
          buffer[j++] = text[i];
     buffer[j] = '\0';
     strcat( buffer, UNDER_OFF );
     return buffer;
}

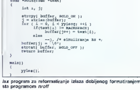
char *bold(char *text)
{
     int i, j;

     strcpy( buffer, BOLD_ON );
     j = strlen(buffer);
     for ( i = 0; i < yyleng; ++i )
          if(text[i] != BACKSPACE)
               buffer[j++] = text[i];
          else
               --j; /* simuliranje BS */
     buffer[j] = '\0';
     strcat( buffer, BOLD_OFF );
     return buffer;
}

main()
{
     yylex();
}
```

lex **program za reformatiranje izlaza dobijenog formatiranjem teksta programom** ***nroff***

Slika 5



Graf prelaza za specifikaciju sa slike 3

" Notacija korišćena za zapisivanje regularnih izraza i njeno značenje je data na slici 4. Napišimo sada modele za naš leksički analizator koristeći regularne izraze:

identifikator	**[A-Za-z]([A-Za-z]|[0-9])***
ceo broj	**[0-9]+**
beline	**[\t\n]+**
sve ostalo	**.|\n**

Jedino odstupanje od prve definicije je u modelu „sve_ostalo". U novoj definiciji vidimo da model zadovoljava svaki string koji se sastoji od samo jednog znaka, i to bilo kojeg.

Postavlja se sledeće pitanje: kako možemo biti sigurni da leksički analizator neće prvu cifru broja "13" prepoznati kao string koji zadovoljava „sve_ostalo"? Odgovor je sledeći: ako je za

dovoljeno više pravila, leksički analizator će odabrati ono kojim se prepoznaje string najveće dužine, ako su stringovi isto dužine biće odabrano pravilo koje koje je prvo napisano u specifikaciji.

U pomenutom slučaju skaner vidi da u ulaznom tekstu nalazi string „1" koji zadovoljava model „sve_stalo" (naravno, ovo pravilo će uvek biti zadovoljeno), ali i da nalazi string „13" koji zadovoljava model „ceo_broj". Pošto je string „13" duži od stringa „1", biće izvršena akcija uz model „ceo_broj". Takođe, da u ulaznom tekstu umesto broja „13" stoji neki jednocifreni broj, na primer „6", opet će biti prepoznat kao „ceo_broj", pošto je ovo pravilo napisano pre pravila za „sve_ostalo".

Regularni izrazi su moćno sredstvo za definisanje nekog skupa stringova. Na primer, bilo koja brojna konstanta se može definisati na sledeći način:

broj -> **[0-9]+(\.[0-9]+)?|\.[0-9]+**

Primeri stringova koji zadovoljavaju ovaj model su:

123 12.3
15 .15

Obratimo pažnju na sekvencu „\." za specificiranje znaka „." (decimalna tačka). Da smo izostavili „\" znak, „." u regularnom izrazu bi bio shvaćen kao „bilo koji karakter osim znaka za novi red", kao što se i vidi iz definicije sa slike 4.

Specifikacijom skanera nismo dobili ništa više od jasnih smernica po kojima treba pisati program koji će ga implementirati.

GRAF PRELAZA

Sledeći korak u pisanju programa je formiranje „grafa prelaza" na osnovu specifikacije. Graf prelaza za specifikaciju sa slike 3 je dat na slici 5. Skaner se na ovaj način predstavlja u vidu „automata" sa konačnim brojem unutrašnjih stanja. Stanja su na grafu predstavljena krugovima, a prelazi između stanja strelicama – potezima koji ih povezuju. Oznake iznad poteza definišu za koji će se ulazni simbol izvršiti taj prelaz.

Početno stanje je stanje „0". Ako u stanju „0" na ulaz dođe slovo, automat će preći u stanje „1", ako dođe cifra, u stanje „2", ako dođe znak beline, u stanje „3", a za sve ostale znake ostaće u stanju „0". Ponašanje u stanju „1" je sledeće: za slovo ili cifru na ulazu automat ostaje u stanju „1", a za sve ostale znake prelazi u stanje „4" kada je niz ulaznih simbola prepoznat kao identifikator. Slično važi i za stanja „2" i „3". Stanja „0", „4", „5" i „6" su nacrtana dvostruko, zato što su to „stanja prihvatanja". Kada se automat nađe u jednom od ovih stanja, znači da je prepoznao odgovarajući token i da treba izvršiti akciju iz odgovarajućeg pravila. Broj stanja prihvatanja je jednak broju pravila u specifikaciji. Akcije napisane pored ovih stanja su akcije iz specifikacije sa slike 3. Posle izvršenja akcije, automat treba „resetovati", tj. treba ga vratiti u početno stanje da bi se prešlo na prepoznavanje sledećeg tokena.

Stanja „4", „5" i „6" su obeležena sa „*" zbog toga što je automat, kada u njih dođe, učitao jedan znak viška – znak koji ne pripada prepoznatom tokenu. Znači, taj znak treba vratiti na ulazni tok, koji posle toga treba da izgleda kao da ovaj znak nije ni bio učitavan. Ovo se u C-u rešava pozivom standardne funkcije **ungetc**.

Kada imamo pred sobom graf prelaza, pisanje programa koji ga implementira je manje-više trivijalan posao. Jedna od mogućih implementacija je data na listingu 1. Primenjen je

REGULARNI IZRAZI

REGULARAN IZRAZ	SKUP STRINGOVA KOJI GA ZADOVOLJAVA
Osnovni regularni izrazi	
x	karakter x
	bilo koji karakter osim NL (\n)
[xyz]	„klasa karaktera": u ovom slučaju bilo koji od karaktera x, y, z
[abj-oZ]	„klasa karaktera" sa intervalom: u ovom slučaju bilo koje od slova a, b, svih slova od j do o i slova Z
[^A-Z]	„negacija klase karaktera": bilo koji karakter IZUZEV velikog slova
[^A-Z\n]	bilo koji karakter IZUZEV velikog slova i NL
Regularni izrazi izvedeni iz osnovnih upotrebom operacija; r i s predstavljaju bilo koje regularne izraze.	
r*	nula ili više stringova određenih sa r
r+	jedan ili više stringova određenih sa r
r?	nula ili jedno r (tj. opciono r)
rs	r praćeno sa s: konkatenacija
r|s	ili r ili s

Operatori su poređani po prioritetu: * ima najviši, a | najniži. Za promenu redosleda računanja određenog prioritetom se koriste zagrade.

Primeri

a|b	string „a" ili string „b"
a|b+	string „a" ili string od jednog ili više slova „b" (npr. „a", „b", „bb", „bbb", ...)
(a|b)*	string sastavljen od nula ili više slova „a" i „b" (npr. prazan string, „a", „b", „ab", „bba", „ababb", ...)

Citiranje karaktera se postiže znakom „\" ili zatvaranjem u dvostruke navodnike. Recimo, specifikacija svih stringova koji se sastoje iz parnog broja znakova „*" se postiže regularnim izrazom (**)* ili („**")* pošto je znak * operator. Takođe, znakom \ se mogu specificirati karakteri svojim heksadecimalnim ASCII vrednostima (npr. \x1b) ili neki kontrolni karakteri na isti način kao u jeziku C (npr. \n i \t).

Osnovni regularni izrazi

najočigledniji, ali ne i najbrži način za simuliranje konačnog automata. Promenljiva „state" pamti tranutno stanje i „switch" kontrolna struktura koja za svako od stanja izvršava odgovarajuće operacije i eventualno menja promenljivu „state". Stanja „4", „5" i „6" su prelazna (automat se u njima zadržava u trajanju jednog koraka) i ne pojavljuju se eksplicitno, već se akcije vezane za njih izvršavaju u okviru stanja „1", „2" i „3".

Program sa listinga 1 čita tekst sa standardnog ulaza i ispisuje niz poruka. Ako ga unesemo u fajl „lexan.c" i prevedemo, a pomenuti paskal program popremimo u fajlu „testlex.pas", naredba

lexan < testlex.pas

će ispisati niz poruka identičan onome navedenom u okviru teksta.

SOFTVERSKA ALATKA FLEX

Kao što vidimo, pisanje programa za leksičku analizu je prilično rutinerski posao, koji za veće specifikacija zahteva puno koncentracije i ne mnogo kreativnosti. Na sreću, postoji računar da obavi dosadan posao – napravljeni su programski alati koji automatski generišu kod skanera na osnovu specifikacije. Jedna od takvih alatki, koja postoji na Unix sistemima kao član standardne biblioteke sistemskih programa, je *lex*. Pošto je, kao i sam Unix, pisana na programskom jeziku C, izvršena su uspešna prenošenja i na druge operativne sisteme. Jedna od verzija *lex*-a koja se može preneti na MSDOS je PD program *flex* razvijen na univerzitetu Berkley. Primeri koji slede su testirani sa *flex*-om. Naravno, zbog visoke međusobne kompatibilnosti svih Unix proizvoda, nema bitnih razlika u odnosu na neku drugu verziju *lex*-a.

Način rada sa *flex*-om je sledeći:

1. Pripremi se specifikacija skenera u fajlu koji se, na primer, zove „lex.l"; ova specifikacija se naziva još i „*lex* program", a sam *lex* „*lex* kompajler". Specifikacija skanera sa slike 3 u vidu *lex* programa je data na listingu 2.

2. Ovaj fajl se propušta kroz *flex* komandom „flex lex.l"; kao rezultat se dobija C izvorni kod skanera u fajlu „lexyy.c"; ovaj fajl sadrži kod funkcije **yylex**, koja je u osnovi ista kao ona data na listingu 1.

3. Fajl „lexyy.c" se prevede C kompajlerom, dobijeni objektni modul se, ako je skaner specificiran kao potprogram, linkuje sa glavnim programom koji će ga pozivati; ako je specificiran da bude samostalan program, pravi se izvršna verzija.

Glavni program „lexmain.c" koji će pozivati generisani skaner izgleda ovako:

```
main()
{
    yylex();
}
```

Posle sledeće sekvence naredbi

flex lex.l
tcc lexmain.c lexyy.c

poziv skanera komandom

lexmain < testlex.pas

bi trebalo da rezultira ispisivanjem istog niza poruka koji je naveden ranije u tekstu, kao kod programa „lexan.c" sa listinga 1.

Pošto smo stekli osećaj o tome kako se radi sa *flex*-om, obratimo pažnju na detalje specifikacije sa listinga 2.

Specifikacija je razdvojena na dva dela, u formi:

definicije
%%
pravila

U delu „definicije" su dodeljena simbolička imena SLOVO i CIFRA odgovarajućim regularnim izrazima koji definišu bilo koje slovo, odnosno bilo koju cifru. Ovakav postupak se zove

„regularna definicija". Definicije imaju oblik

```
ime                  definicija
```

pri čemu „ime" mora započeti od prve kolone. Početkom definicije se smatra prvi neblanko znak iza imena. Na ove definicije se pozivmo u pravilima tako što navedemo „ime" okruženo vitičastim zagradama u u okviru regularnog izraza, na primer „{CIFRA}". Kada se naiđe na njega u pravilu, string „{ime}" se zamenjuje stringom „(definicija)". Ovo je u suštini postupak makro zamene, kao kod rada sa C preprocesorskom direktivom **#define**.

Deo „pravila" se sastoji od niza pravila oblika

```
model                akcija
```

Model se specificira proširenim skupom regularnih izraza. Ovo uključuje sve regularne izraze na osnovu definicije na slici 4, s tim što su podržane i neka proširenja koja zbog prostora nećemo navoditi.

Akcije se zadaju u vidu proizvoljnih naredbi programskog jezika C. Model mora otpočeti od prve kolone. Početkom akcije se smatre prvi neblanko znak posle regularnog izraza i proteže se do kraja linije. Izostavljanje akcije ima značenje „ne radi ništa" i u tom slučaju se pronađeni token jednostavno odbacuje. Ako se akcija prostire na više redova, mora biti oivičena na vitičastim zagradama, kao blok u C programu. Na primer

```
hello                { i++;
                     printf(„Zdravo \n");
                     }
```

Za tekst koji ne može prepoznati ni jednim pravilom, preduzima implicitna akcija, koja ga samo ispisuje na standardni izlaz. Sledeći skaner će prepisati svoj standardni ulaz na standardni izlaz uz zamenjivanje svake pojave stringa „ime" stringom „Petar"

```
%%
ime                  printf(„Petar");
```

Vrednost stringa u ulaznom tekstu koji biva prepoznat, i njegove dužine su dostupni u globalnim promenljivama

```
char *yytext
int yyleng
```

Primer upotrebe promenljive **yytext** je se vidi u specifikaciji na listingu 2.

JEDAN PRAKTIČNI PROBLEM

Sledeći primer prikazuje upotrebu leksičkog analizatora generisanog *flex*-om za rešavanje jednog praktičnog problema. Prvo ćemo izložiti sam problem.

Uz većinu Unix softvera (pa i uz sam *flex*) dokumentacija se isporučuje u obliku „izvornih" fajlova, koji se u upotrebljiv oblik prevode programom za formatiranje teksta *nroff*. Ovakav program kao rezultat svog rada formira novi fajl, koji se sastoji od teksta i kontrolnih kodova, i koji treba poslati direktno na printer.

Pošto se teži nezavisnosti od printera na kojem će se štampanje obaviti, od kontrolnih karaktera se koristi jedino „backspace" (BS, ASCII vrednost 8). Štampanje u „bold" i „underline" režimu se obavija korišćenjem BS karaktera na način koji je prikazan na primeru. Tekst „abc" se štampa „bold" sledećom sekvencom

a BS a BS a BS a b BS b BS b BS b c BS c BS c BS c

a „underline" sledećom

_ BS a _ BS b _ BS c

Pošto je štampanja ovakvih sekvenci na matričnom printeru izuzetno sporo i naporno za printer, bilo bi zgodno prevesti ih u oblik u kojem se „bold" i „underline" efekat postiže upotrebom **escape** sekvenci. Znači navedene sekvence trebalo zameniti sledećim

"bold on" a b c „bold off"
"underline on" a b c „underline off"

„bold on", „bold off", „underline on" i „underline off" su **escape** sekvence kojima se vrši promena režima štampanja.

Program koji ćе ova zamene da obavlja se može lako generisati *flex*-om. Odgovarajuća specifikacija je data na listingu 3. Pretpostavimo da se nalazi u fajlu „reformat.l" i da smo izvršili sledeću sekvencu naredbi

```
flex -t reformat.l > reformat.c
tcc reformat.c
```

Opcija „-t" nalaže *flex*-u da rezultat svog rada pošalje na standardni izlaz umesto u fajl „lex.yy.c". Vidimo da je *flex* ovaj put generisao samostalan program, a ne potprogram kao u prethodnom primeru. Zašto je tako, biće uskoro objašnjeno. Štampanje dokumenta „flex.man" koji je dobijen kao izlaz *nroff*-a obavljamo naredbom

```
reformat < flex.man > PRN
```

Na specifikaciji sa listinga 3 vidimo nova pojedinosti. Potpun izgled specifikacije je sledeći

```
definicije
%%
pravila
%%
korisnički kod
```

Tekst koji se nalazi u delu „korisnički kod", koji počinje iza drugog „%%", doslovno se kopira u generisani leksički analizator. U ovom delu se pišu naredbe funkcija koje se pozivaju u akcijama. U ovom slučaju vidimo da je u njemu definisana i funcija **main**, koja poziva leksički analizator **yylex**. To je razlog što je generisan kod samostalanog programa.

Takođe, na listingu 3 vidimo da se u delu „definicije" nalaze prototipovi funkcija koje se pozivaju u okviru akcija „underlined" i „bold". Prototipovi se razlikuju od regularnih definicija po tome što su uvučeni (*indented*). Na ovaj način se mogu definisati i globalne promenljive koje će biti dostupne i u funkciji **yylex** i u funkciji **main**. To je prikazano u sledećem primeru, kojim se generiše program za brojanje reči i redova u tekstu

```
int chars=0,
int words=0;
%%
\n                   ++lines, ++chars
                     ++chars;
%%
main()
{
yylex();
printf("Text has %d lines and %d
characters \n", lines, chars);
}
```

Iz izloženog se vidi da je *flex* jako zgodna alatka za brzo i jednostavno generisanje programa za manipulaciju tekstom. Mogućnosti *flex*-a izložene u primerima su daleko od punе snage. Sa njima je najbolje upoznati se kroz postepeno čitanja dokumentacije i eksperimentisanje.

Ova alatka će dobro doći svakom ko se intenzivno bavi programima za obradu teksta i, naravno, dizajnom kompajlera i interpretatora. Ušedeće mu puno vremena i živaca, a programi će biti izuzetno efikasni. Međutim, i onaj ko se ne bavi time može povremeno iskoristiti *flex* za rešavanje nekog problema.

Recimo, da li ste se nekad sreli sa problemom zamene jednog stringa drugim u tekstu veličine 500 K? Teško je naći editor koji će se izboriti sa ovako velikim tekstom. Zamislite još da je potrebno vršiti i neke složenije manipulacije. Pomoću *flex*-a možete lako i brzo da napravite program koji će takav zadatak obaviti. Veličina ulaznog fajla mu neće smetati, a osim toga, algoritam po kojem će raditi generisani skaner je optimalan, tako da će i brzina rada biti velika.

I na kraju, *flex* ne zauzima više od 100 K na vašem disku.

Sve u svemu, razloga da nabavite i naučite da koristite ovu lepu i u svetu MS DOS a relativno nepoznatu programersku alatku, je više nego dovoljno.



PLANIRANJE BAZE

Složen program nije prost zbir više jednostavnijih modula – pisanje aplikacije od nekoliko desetina hiljada programskih redova zahteva mnogo detaljniju i pažljiviju analizu problema. U slučaju baza podataka, planiranje strukture programa je samo jedan korak u radu: preostaje i planiranje strukture baze podataka.

Nenad Batoćanin

Pisanje velikih aplikacija uglavnom je povezano sa nizom problema koji se ne pojavljuju u manjim programima. Između ostalog, potrebno je pažljivo osmisliti aplikaciju kako bi se mogli koristiti overleji, mora se izbeći dupliranje podataka u datotekama itd. Problem je što je svaka aplikacija, pa i velika, na početku razvoja – mala, pa se u početku neće ni primetiti ako je neki deo „sklepan" na brzinu. Međutim, kako kod raste, problemi nepogrešivo nalaze pukotine u konstrukciji programa i nezadrživo krive celu građevinu. Krpljenje rupa samo još više nakrivljuje celu konstrukciju i tada preostaje jedino rešenje – ponovno pisanje iz početka. Naravno, bolje je sprečiti nego lečiti: ako pre pisanja programa odvojite nekoliko sati na čisto planiranje, situacija postaje daleko bolja. U ovom tekstu ćemo detaljnije razmotriti planiranje baze podataka za *Clipper* aplikacije. Problemi nikako nisu direktno vezani za *Clipper* – svi pobrojani problemi javljaju se u kreiranju iole veće baze podataka. To znači da naredni tekst može biti interesantan i onima koji ne koriste *Clipper* već neku drugu relacionu bazu podataka.

U svetu postoji ogroman broj knjiga koje obrađuju ovaj problem. I kod nas je moguće pronaći izvestan broj ovakvih knjiga, ali su one uglavnom ili previše uopštene za svakodnevnu upotrebu ili sadrže opis nekog konkretnog rešenja koje vama ne odgovara. Naredni tekst je pokušaj da se na jednom mestu ukratko opišu osnovne stvari neophodne za planiranje baze podataka.

IDENTIFIKACIONI BROJ SLOGA

Vrlo često se sreću aplikacije u kojima se na početku kreiraju potrebne datoteke, ako ne postoje. To omogućuje da program isporučite kao jedinstvenu **EXE** datoteku, koja kasnije po potrebi kreira ostale datoteke. Uglavnom je stvar ličnog ukusa da li će se raditi ovako ili će se uz program isporučivati i prazne datoteke. Međutim, veoma je važno da autor programa ima tačan uvid u to koje sve datoteke postoje u sistemu i njihove međusobne odnose. Zato je preporučljivo da paralelno sa razvojem aplikacije vodite spisak datoteka sa detaljnim opisom strukture i **komentarima** za svako polje pojedinačno. To će vam kasnije uštedeti sate i sate „provaljivanja" i lutanja po sopstvenom kodu. Nedostatak programa koji automatski kreira potrebne datoteke je to što procedure za kreiranje morate paralelno ažurirati sa svim izmenama baze podataka, što ostavlja prostor za veoma teško uočljive greške.

Jedan od važnih faktora pri planiranju strukture DBF datoteke (tabele) je izbor izraza po kojima će se vršiti indeksiranje (tzv. **ključeva**). Šta ima odlučujuću ulogu u tome? Najvažniji kriterijum je način kasnije upotrebe te datoteke. Nevolja je u tome što je nemoguće tačno unapred predvideti kako će se jedna datoteka koristiti. Ipak, ovde je potrebno „pogledati u budućnost" bar donekle – što budete precizniji, kasnije ćete imati manje posla.

Svaka datoteka se može indeksirati po više izraza, ali pri tome važi pravilo da uvek treba imati jedan na osnovu koga ćete moći jednoznačno da identifikujete slog. Ako je datoteka „centralna" u sistemu, važnost ovog pravila je još veća. Na primer, ako vodite evidenciju radnika u nekom preduzeću, sasvim je sigurno da će vam pre ili kasnije zatrebati da jednoznačno pronalazite radnika. Postavlja se pitanje šta izabrati za ključ. Ime i prezime je nezgodno – ako se i zanemare nepogodnosti tipa promene prezimena i imena, postojanje dve osobe sa istim imenom u slučaju veće količine podataka zaista nije nikakvo čudo. Ostali mogući podaci su takođe „klimavi" – identifikacioni broj za koga se u firmi kunu da je jedinstven za svakoga pre ili kasnije ispadne vrlo nejedinstven itd.

Zato se za svakog radnika (i uopšte svaki slog) uvodi **identifikacioni broj** (u daljem tekstu ćemo ga obeležavati sa ID). To je ceo broj na osnovu koga će se na jedinstven način „adresirati" svaki slog. Velika greška je da se za ovaj broj upotrebi redni broj sloga – svaki slog datoteke ima jedinstven broj koji se dobija funkcijom **RecNo()**, ali stvar je u tome što se taj broj ne menja samo kad je datoteka statična. Problem nastaje sa brisanjem slogova. Na primer, ako negde zapišete „radnik broj 120 je dobio platu od 20000 dinara", rezultat može postati netačan posle prvog brisanja: ako se izbriše neki slog sa rednim brojem manjim od 120, svim slogovima sa rednim brojem većim od 120 smanjiće se redni broj za 1.

Sledeća česta greška je da se uvede polje u kome se pamti identifikacioni broj, ali da se kod upisa novog sloga identifikacioni broj određuje po formuli

<ID poslednjeg sloga> + 1

U čemu je ovde problem? Program će sasvim lepo raditi sve dok datoteka ne pretrpi malo veće izmene (brisanje i upisivanje slogova), a onda će doći do veoma ozbiljnih grešaka. Na primer, poslednji upisan slog ima **ID = 200**. Ako se sad ovaj slog izbriše, pa kasnije doda novi slog, njegov ID biće (najverovatnije) isto 200! Sada će se podaci za novi slog preklapati sa ranije unetim podacima za stari slog sa identifikatorom 200 i etc potencijalnog izvora problema. Pravo rešenje je uvođenje **jedinstvenog** identifikacionog broja za svaki slog, bez obzira na njegovo prisustvo u sistemu. Za određivanje takvih brojeva može nam poslužiti funkcija **NextID** sa slike 1. Ona pri svakom pozivu daje celi broj za jedan veći od prethodnog. Tekući broj se čuva u datoteci koju navodite kao argument funkcije. Preporučljivo je da naziv datoteke u kojoj se pamti ID sloga bude isti kao i naziv odgovarajuće DBF datoteke. Svaki upis novog sloga nadalje počinje sa

```
USE Proba
…
APPEND BLANK
REPLACE
proba->ID WITH NextID ("proba.id")
```

Tako će svaki slog imati svoj broj koji se neće mešati sa novim ili starim brojevima drugih slogova. Jedini nedostatak ovog rešenja je što će pre ili kasnije doći do „prepunjenja": navedena funkcija vraća brojeve do 99999, tako da se mora predvideti *periodično* anuliranje ovih brojača.

INDEKSIRANJE, PRETRAŽIVANJE, IZDVAJANJE

Kada se vrši izbor ključeva po kojima će se vršiti indeksiranje, gotovo uvek treba odabrati ID broj kao glavni ključ. Važno je uočiti da indeksiranjem obično rešavamo i sortiranje podataka: vrlo retko će biti potrebno da se slogovi fizički premeštaju po nekom zadatom redosledu. Mnogo bolje (i brže) rešenje je indeksiranje, jer se indeksiranjem postiže da korisnik vidi istu sliku kao da je datoteka sortirana, ali sa daleko većom fleksibilnošću. Pomenuću



Slika 1

```
*********************
*
* DATUM:  6. 7.1990.
*
* NAZIV: S := NextId (FileID)
*
* OPIS:  Vraca sledeci broj na osnovu datoteke FileID
*
*
*********************

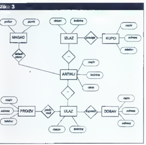
FUNC NextId (FileID)
   LOCAL ID, fp, Buff

   // Ako datoteka ne postoji, ID == 1
   ID := 1
   // Datoteka postoji...
   IF File (FileID)
      fp := FOpen (FileID, 0)
      Buff := Space(5)
      FRead (fp, @Buff, 5)
      max := Val (Buff) + 1
      FClose (fp)
   END IF

   // Svejedno kreiraj novu datoteku i upisi tekuci...
   fp := FCreate (FileID)
   FWrite (fp, str(ID,5,0))
   FClose (fp)
RETURN ID
```

Određivanje novog identifikacionog broja za slog DBF datoteke

Slika 3



Primer dijagrama *objekti-veze* za jedno trgovinsko preduzeće

samo da je ovako moguće skoro trenutno menjati nekoliko različitih redosleda i slično. Ostali ključevi se određuju bez nekih određenih pravila, po ukazanoj potrebi.

Datoteka se indeksira zbog ubrzanja **pretraživanja** datoteke. To je u stvari naziv za traženje **jednog** sloga po zadatom uslovu. Za pretraživanje se može koristiti više postupaka, počev od naredbe **SEEK**, preko **LOCATE** sve do jednostavnog „pregleda" svih slogova.

Naredba **LOCATE** služi za traženje sloga koji zadovoljava neki zadati uslov. Na primer, naredba **LOCATE FOR ID == 200** traži prvi slog čije polje ID ima vrednost 200. Ako takav slog ne postoji funkcija **Found()** imaće vrednost .**F**. Naredba **LOCATE** redom za svaki slog datoteke računa vrednost navedenog izraza i zaustavlja se kad nađe slog koji zadovoljava izraz (izraz ima vrednost .T.) ili kad dođe do kraja datoteke.

Naredba **SEEK** traži slog po vrednosti zadatog ključa. Na primer, ako je datoteka indeksirana po polju **ID**, slog sa ID = **200** traži se naredbom **SEEK 200**. Početnici obično ne uočavaju bitnu razliku između ova dva načina pretraživanja. Traženje sloga naredbom **SEEK** je neuporedivo brže, jer se ne računa logički izraz redom za svaki slog datoteke, nego se pretraživanje vrši korišćenjem specijalne strukture koja se naziva **B-stablo** i koja se kreira kreiranjem indeksa. Čemu onda služi naredba **LOCATE**, zapitaćete se. Zakon održanja ovde je vrlo očigledan: naredba **SEEK** mnogo brže traži slog, ali zato može tražiti samo po **unapred** specificiranom izrazu po kome je izvršeno indeksiranje, dok naredba **LOCATE** omogućava da uslov za traženje bude proizvoljno komplikovan i da se može zadati čak i za vreme izvršenja programa. Tako u opštem slučaju važi da naredbu **SEEK** uvek treba upotrebljavati kad je moguće unapred znati izraz po kome će se vršiti traženje – tako će vaš program znatno brže raditi.

Sledeća važna razlika je između **pretraživanja** i **izdvajanja**. Kao što smo videli, pretraživanje je nalaženje sloga po nekom kriterijumu, a **izdvajanje** je nalaženje **svih** slogova koji zadovoljavaju zadati kriterijum. Za izdvajanje služi naredba **SET FILTER** kojom se zadaje logički uslov koji je tačan za slogove koje treba izdvojiti. Na primer, **SET FILTER TO plata > 10000** izdvaja sve slogove kod kojih je vrednost polja **plata** veća od 10000. Često se programeri žale kako je naredba **SET FILTER** katastrofalno spora. Pri tom se mora imati u vidu kako ta naredba radi. Naredba **SET FILTER** pri pomeranju pokazivača sloga, pre nego što dozvoli pristup slogu, računa izraz zadat sa **SET FILTER** za tekući slog. Ako je taj logički izraz tačan, pristup slogu se dozvoljava, a u suprotnom se postupak ponavlja za sledeći slog, sve dok se ne pronađe slog koji zadovoljava uslov. To znači da je datoteka samo prividno „smanjena" na slogove koji zadovoljavaju uslov – sva obrada se **uvek** vrše sa celom datotekom. Odatle i poznati efekat kad korisnik u **BROWSE** pregledu pritisne taster sa strelicom na dole, a program „zaglavi" na neko vreme – za to vreme se u stvari računa vrednost logičkog izraza za svaki slog od tekućeg do kraja/početka datoteke. Ako takvih slogova ima puno, duže će trajati „zaglavljivanje" računara. **DBEdit** funkcija je bila vrlo nepogodna za rešavanje ovog problema, dok novi **TBrowse** sistem omogućava skoro trivijalno rešenje. Dovoljno je pre početka pregleda zapamtiti prvi i poslednji **logički** slog datoteke i kasnije metode (funkcije) za pomeranje gore i dole zvati samo ako slog nije prvi/zadnji. Na primer:

CASE Ch == K_DOWN .AND. RecNo() <> Last
TB:Down()

Male komplikacije nastupaju ako se za vreme pregleda menja datoteka, ali se i to može rešiti dosta jednostavno. Naredbu **SET FILTER** treba koristiti što ređe, jer je prilično „kruta" za upotrebu. Mnogo je preporučljivije na neki način markirati slogove i uvek raditi sa celom datotekom. Na taj način je osigurana potpuna kontrola.

Važno je uočiti da naredba **SET FILTER** u suštini radi mnogo sporije od **LOCATE**, jer se u opštem slučaju traži računanje mnogo više izraza. Zato je bolje uvek pretraživati nego izdvajati – ako korisniku ne trebaju svi podaci koji zadovoljavaju uslov ili postoji samo jedan takav podatak, uvek je bolje koristiti naredbu **LOCATE** nego **SET FILTER**.

POVEZIVANJE VIŠE DATOTEKA

Prvi korak ka „ozbiljnoj" bazi podataka u vašem programu svakako je povezivanje više DBF datoteka određenim relacijama. Navešćemo jedan jednostavan primer. Pretpostavimo da vodimo evidenciju dolazaka radnika nekog preduzeća na posao: za svakog radnika treba voditi evidenciju kad je došao i kad otišao, po danima. Prvi korak je, svakako, kreiranje DBF datoteke koja će sadržati spisak radnika. Neka svaki slog sadrži i ID broj, kao što je prethodno objašnjeno. Pošto po logici stvari podatke o dolasku treba nekako „vezati" za slog radnika, početnici neretko pokušavaju da svakom slogu dodaju 30 i više polja, po jedno za svaki dan u mesecu. Takvo rešenje je vrlo loše – čak i da se pretpostavi da nije potrebno voditi evidenciju više od 30 poslednjih dana, računanje sa ovakvom strukturom je izuzetno teško. Daleko bolje rešenje je izdvajanje podataka o dolascima u posebnu datoteku. Postavlja se pitanje koja polja treba kreirati u toj datoteci. Naravno, polja u kojima će se pamtiti vreme dolaska i vreme odlaska, i eventualna dodatna polja (za rada ...) nisu sporna. Problem je u nečem drugom. Naime, kako označiti na kog radnika se odnosi podatak? Najjednostavnije (i najlošije) rešenje je uvesti polje za ime i prezime i u svakom slogu datoteke **DOLASCI** pamtiti ime i prezime radnika. Mada vam se u prvom trenutku može učiniti da je ovo najprirodnije rešenje, daleko ste od istine. Pogledajte samo ovaj spisak potencijalnih problema:

- Negde na sredini evidencije radniku je promenjeno ime, pa će se tako podaci do tog dana „izgubiti".
- Postoji više radnika sa istim imenom.
- Firma ima puno radnika i datoteka **DOLASCI** brzo raste: ako za svaki slog dodate polje za ime i prezime (oko 30 simbola), za godinu dana za evidenciju 1000 radnika datoteka će biti najmanje 30·1000·300 = 9000000, a to je više od 9MB.

Mnogo je bolje da umesto imena i prezimena radnika kao identifikaciju uvedemo baš ID sloga radnika iz datoteke **RADNICI**. Tako će se

Slika 2



Povezivanja dve datoteke (RADNICI i DOLASCI)

podaci u datoteci **DOLASCI** jednoznačno identifikovati na osnovu ID a radnika uvek možemo pronaći o kom se radniku radi

Sledeći korak je **povezivanje** ove dve datoteke. Za to služi naredba **SET RELATION** koju *Clipper* programeri začuđujuće retko koriste. Mada u početku može izgledati da je ova naredba komplikovana, ona je u stvari veoma jednostavna. Njena sintaksa je **SET RELATION TO <izraz> INTO <oblast>**. <izraz> je proizvoljan izraz u tekućoj radnoj zoni a **<oblast>** je radna zona sa kojom se vrši povezivanje. Dejstvo je jednostavno: uvek kad se pomeri pokazivač sloga u osnovnoj zoni, izvršće se naredba **SEEK <izraz>** u zoni navedenoj posle **INTO**. Tako će se pokazivač sloga u drugoj datoteci automatski pomeriti na odgovarajući slog. Naravno, u zoni u kojoj je otvorena druga datoteka mora postojati aktivan indeks kako bi naredba **SEEK** ispravno radila. U našem slučaju za povezivanje datoteka **RADNICI** i **DOLASCI** treba izvršiti sledeći niz naredbi

```
USE Radnici INDEX RadniciID NEW
USE Dolasci NEW
SET RELATION TO dol_rad INTO Radnici
```

Pretpostavimo da je datoteka radnika indeksirana po ID broju, a da u datoteci **DOLASCI** polje **dol_rad** predstavlja ID radnika na kog se odnosi podatak. Primera radi, neka u datoteci radnika postoji slog sa podacima **ime** = **„Mitar Mitrović"** i **id** = **21**. Tada će svi slogovi u datoteci **DOLASCI** koji se odnose na tog radnika imati polje **dol_rad** = **21**. Kada se pokazivač sloga u datoteci **DOLASCI** nekom (proizvoljnom) naredbom pomeri na slog u kome je polje **dol_rad** = **21**, u zoni **Radnici** izvršiće se naredba **SEEK 21**, pa će tekući slog u datoteci **RADNICI** postati upravo onaj koji nam treba (slika 2), tj. izraz **radnici->ime** imaće vrednost "**Mitar Mitrović**". Sada se relativno lako mogu dobijati raznovrsni podaci. Na primer, spisak svih radnika koji su došli određenog dana na posao dobija se naredbom

```
LIST
radnici->ime FOR dol_datum == Date()
```

Primetite da je ova veza jednosmerna i to od datoteke **DOLASCI** prema **RADNICI**. To znači da pomeranje pokazivača u datoteci **RADNICI** ne menja tekuću poziciju pokazivača sloga u datoteci **DOLASCI**. Kako se određuje smer povezivanja razmotrićemo malo kasnije. Za sada ćemo napomenuti da se mogu dobiti i podaci u „obrnutom" smeru. Na primer, spisak svih datuma dolazaka jednog radnika dobija se naredbama

```
SELECT Radnici
// Pomeri se pokazivač na radnika
// za kog su potrebni podaci
id_radnika = rad_id
SELECT Dolasci
LIST
dol_datum FOR id_radnika == dol_rad
```

Slika 4



Moguće veze između tabela

Naravno, za dobijanje ovih podataka nije se koristila veza uspostavljena naredbom **SET RELATION** – ovakav izveštaj se može dobiti uvek kada su datoteke organizovane na prethodno opisani način.

Na sličan način se može povezivati i više datoteka. Na primer, ako je evidencija dolazaka realizovana na prethodno opisani način, a slično se vodi evidencija o radnim zadacima koje je radnik određenog dana imao, naredbe koje datoteku **DOLASCI** povezuju sa datotekama **RADNICI** i **ZADACI** može biti

```
SELECT Dolasci
SET RELATION TO dol_rad
INTO Radnici, dol_zad INTO Zadaci
```

Sad će svako pomeranje pokazivača sloga u datoteci **DOLASCI** prouzrokovati pomeranje pokazivača u datotekama **RADNICI** i **ZADACI**. Ne postoji nikakva prepreka ni za višestruko povezivanje datoteka: datoteka **RADNICI** dalje može biti povezana sa nekom drugom datotekom itd. Tako se datoteka **DOLASCI** može posredno povezati sa velikim brojem drugih datoteka. Jedino ograničenje (osim broja veza) je ciklično povezivanje. To bi značilo da je datoteka direktno ili indirektno povezana sa samom sobom, što bi prouzrokovalo beskonačno ponavljanje naredbe **SEEK** (svako pomeranje pokazivača sloga u zoni bi indirektno ili direktno opet prouzrokovalo pomeranje pokazivača u toj zoni).

VRSTE VEZA

U planiranju baze podataka veoma je važno dobro organizovati veze između tabela. Zato ćemo malo detaljnije razmotriti povezivanje tabela. Ma kako izgledali raznovrsni slučajevi povezivanja, sve mogućnosti se svode na sledeće tri (slika 4):

- svakom slogu jedne datoteke odgovara tačno jedan slog druge datoteke. Ova veza se zove **1-1** veza. Na primer, svaki radnik ima tačno jedan broj na platnom spisku i svaki broj na platnom spisku odgovara tačno jednom radniku
- svakom slogu datoteke odgovara više slogova druge datoteke, a svakom slogu iz druge datoteke odgovara tačno jedan slog iz prve datoteke. Ova veza se zove **1-N** veza. U prethodnom primeru, svakom radniku odgovara više slogove u datoteci **DOLASCI**
- svakom slogu prve datoteke odgovara više slogova druge datoteke i svakom slogu druge datoteke odgovara više slogova prve datoteke. Ova veza se zove **M-N** veza. Na primer, ako projekat radi više radnika, svaki radnik može učestvovati u jednom ili više projekata, a projekat rade jedan ili više radnika

Bez obzira na jednostavnost ovih osnovnih veza, cela baza podataka može sadržati **veoma** komplikovanu strukturu. Zato je pogodno imati na neki način predstavljenu strukturu zbog lakšeg bržeg snalaženja i boljeg uvida. Jedna od standardnih mogućnosti za predstavljanje strukture baze podataka je upotreba dijagrama **objekti-veze**. To je formalni način za grafičko predstavljanja strukture baze podataka. Ovakav pristup je mnogo bliži formalnoj reprezentaciji relacione baze podataka nego *Clipper* ovom sistemu DBF datoteka, ali i ovde može veoma korisno poslužiti. Primer takvog dijagrama za neko trgovinsko preduzeće prikazan je na slici 3. Pravougaonicima su predstavljeni **objekti** (to su **ARTIKLI**, **IZLAZ** ... i predstavljeju u opštem slučaju DBF datoteke). Elipse predstavljaju **atribute** objekata (to su polja unutar DBF datoteke). Na primer, atributi objekta **ARTIKLI** su **naziv**, **količina** ... Romboidi predstavljaju **veze** između objekata. Veze mogu, ali ne moraju predstavljati posebne DBF datoteke. Krajevi veze se mogu označavati simbolima > i >> kako bi se znalo o kom se tipu veze radi. Na primer, veza na čijem je jednom kraju oznaka < a na drugom >> označava vezu **1-N**. Pošto atributi mogu biti veoma brojni, slika se može znatno pojednostaviti ako se oni uklone (na datoj slici takođe nisu prikazani svi elementi). S obzirom da za strukturu nisu previše važni, u slučaju malo komplikovanije slike atribute obavezno treba izostaviti.

Slika 5



***N-M* veza između dve datoteke (AUTORI i KNJIGE)**

Ovakva slika je posle kraćeg privikavanja izuzetno pregledna i zahvalna za izmene. Vrlo dobar pristup je da za svaku složeniju aplikaciju paralelno ažurirati i jedan ovakav crtež.

1-1 veze su najjednostavnije i veoma lako se implementiraju. Na primer, ako postoje posebne datoteke sa spiskom radnika i njihovim šiframa za pristup računarskom sistemu, veza između ove dve datoteke je **1-1**. Ovde je umesno postaviti pitanje **zašto** dve datoteke? Jer ako je veza jednoznačna, dovoljno je u datoteku **RADNICI** dodati još nekoliko polja za evidenciju šifara i tako izbeći komplikacije. Ovo uvek treba raditi kad je to moguće – bez obzira na brzinu izvršavanja naredbe **SEEK**, uvek je bolje imati podatke na jednom mestu. Problem su **1-1** veze koje nisu **potpune**. To znači da nisu **baš** svi elementi povezani. Na primer, čistačica u preduzeću ne radi na računaru i nema svoju šifru. Može se desiti i obrnut slučaj: neki radnik je prešao u drugu firmu, a i dalje postoji šifra na sistemu sa kojom je on radio. Očigledno je da u oba slučaja „nedostaje" po jedan element sa jedne strane relacije. To se u dijagramima obično označava simbolom **o** sa odgovarajuće strane i kaže se da je veza **parcijalna**. Ovakve relacije nešto komplikuju realizaciju. Naime, ako smo i radnike i šifre smestili u jednu datoteku, javiće se problemi kod nedostatka informacija sa jedne strane. Na primer, ako čistačica nema svoju šifru, može se staviti neka specijalna oznaka koja označava da ta šifra ne postoji (npr. blanko simboli). Ali šta ako je veza parcijalna sa druge strane? Jer ako postoji šifra, a ne postoji radnik, nije nimalo zgodno njegovo ime i prezime ispuniti blan-

ko simbolima – ime i prezime radnika se vero vatno koristi i za druge potrebe. Na primer, nije ni malo zgodno u spisku radnika izlistati i neko liko praznih redova!

U ovom slučaju tabela šifara treba da sadrži ID radnika i odgovarajuću šifru. Datoteku treba indeksirati po ID-u radnika. Tako će sledeći niz naredbi uspostaviti vezu RADNICI–ŠIFRE:

USE Sifre INDEX Sifre NEW
USE Radnici INDEX Radnici NEW
SET RELATION TO ID INTO Sifre

Ako radnik nema šifru, izraz **sifre->(Found())** biće postavljen na .F. Obrnuto traženje je veoma jednostavno: ako je pokazivač sloga datoteke **SIFRE** pozicioniran na nekoj šifri, radnik koji ima tu šifru dobija se sa

SELECT Radnici
SEEK Sifre->sif_rad

gde je **sif_rad** polje datoteke **SIFRE** sa ID-om radnika.

Malo veću komplikaciju predstavlja slučaj kada je **1-1** veza parcijalna sa obe strane. Evidencija stanodavaca i podstanara predstavlja takav primer. Svaki stanodavac može izdati stan tačno jednom podstanaru i jedan podstanar može uzeti samo jedan stan (primera radi smo uprostili opšti slučaj!), ali stanodavac **ne mora** izdati stan niti podstanar mora imati „svog" stanodavca. Rešenje u vidu logičkog polja **zauzet** koje pokazuje da li objekat ima odgovarajućeg partnera je primenljivo, ali je daleko od savršenog. Mnogo je bolje napraviti novu datoteku u kojoj će se pamtiti veze između objekata u dvema datotekama. Za naš primer to bi bila datoteka **STAN_PODS** koja bi imala polja ID stanodavca i ID podstanara. Svako nalaženje stana bi upisali kao nov slog u ovu datoteku. Sada je jednostavno dodati nove atribute ove veze: datum sklapanja ugovora, cena pri posredovanju itd. Primetite da ovde možemo dobijati neke podatke koje ranije nije bilo moguće dobiti. Na primer, moguće je dobiti podatke o vezi koja je važila ranije (podstanar je stanovao u nekom stanu, pa se iselio u drugi).

Slučaj **1-N** veza između tabela je daleko češći. Prethodni primer za evidenciju dolazaka radnika na posao predstavlja upravo takav slučaj: svakom slogu u datoteci **RADNICI** odgovara više slogova u datoteci **DOLASCI**. Ova veza standardno realizuje tako što se u datoteci sa strane **N** kreira polje koje sadrži identifikator odgovarajućeg sloga datoteke sa strane **1**, kako je malopre objašnjeno. I ovde se mali problemi javljaju kad je veza parcijalna sa strane **1**. To znači da može postojati slog koji nema pridruženih slogova u drugoj datoteci. Na primer, tek primljen pripravnik nema nijednog podatka o dolascima na posao. I pored to ga nema neke posebne potrebe za izdvajanjem ovakve relacije u posebnu datoteku.

Veza **N-M** se dosta često javlja u praktičnim problemima. Na primer, odnos između autora i knjiga je takav: svaki pisac može napisati više knjiga, ali i svaku knjigu može napisati više autora. Ovde je, sem u specijalnim slučajevima, neophodno koristiti posebnu datoteku za pamćenje relacije. Mnogi programeri pribegavaju polovičnim rešenjima, na primer, uvode više **CHARACTER** polja ili memo polje za autore. Ovo je veoma loše zbog kasnije nadgradnje i zbog toga što se uvećava količina podataka koji se ponavljaju. Tako se ime nekog poznatog pisca može ponoviti mnogo puta, što nepotrebno uvećava bazu i otežava kasnije izmene. Pretraživanje tipa „nađi sve knjige koje je napisao pisac XX" je skoro nemoguće za veće baze podataka.

U slučaju **N-M** veze treba kreirati posebnu datoteku koja sadrži parove ID-ova slogova datoteka koje se povezuju. Na primer, ako je Tolstoj (ID = 243 u datoteci **AUTORI**) napisao „Rat i mir" (ID = 5789 u datoteci **KNJIGE**), onda će u datoteci **NAPISAO** postojati slog **(243, 5789)**. Naravno, po potrebi se mogu dodati i neki atributi te veze – godina kad je knjiga napisana i sl. Ova nova datoteka se obično povezuje sa datotekama **AUTORI** i **KNJIGE** na sledeći način:

USE Knjige INDEX Knjig_ID NEW
USE Autori INDEX Autor_ID NEW
USE Napisao NEW
SET RELATION TO nap_autor
INTO Autori, nap_knjiga INTO Knjige

Sad će svako pomeranje sloga u datoteci **NAPISAO** paralelno pomerati pokazivače u datotekama **AUTORI** i **KNJIGE** (slika 5). Standardni podaci koji se traže od ovakve veze su „sve knjige od jednog autora" i „svi autori jedne knjige". Ako je potrebno izlistati sve parove (pisac, knjiga), to se jednostavno radi sledećom naredbom:

SELECT Napisao
LIST ALL Autori->ime, Knjige->naslov

Sve knjige od jednog autora se dobijaju sledećim naredbama:

SELECT Napisao
LIST ALL Knjige->naslov
FOR Autori->napisao == „Lav Tolstoj"

a svi autori jedne knjige:

SELECT Napisao
LIST ALL Autori->ime
FOR Knjige->naslov == „Rat i mir"

Pri povezivanju datoteka se ravnopravno mogu koristiti sve ostale naredbe i funkcije za rad sa bazom podataka: **SET FILTER**, **INDEX**... Ako su veze složenije, može doći do toga da maksimalni broj otvorenih datoteka specificiran direktivom **FILES=NN** u **CONFIG.SYS** bude premali. U tom slučaju treba redukovati broj aktivnih veza i otvorenih datoteka u jednom trenutku i aktivirati ih po potrebi. Najbolje je aplikaciju podeliti na logičke celine (module) i na početku svakog modula otvoriti potrebne datoteke, indekse i postaviti relacije, a na kraju sve datoteke zatvoriti. Time povećavate nezavisnost pojedinih modula – kada kasnije budete dodavali novi modul, nećete morati da pregledate koje sve otvorene datoteke ostavljaju „u nasleđe" ranije napisani moduli.

Često se desi da je broj podataka u DBF datoteci veoma mali. U takvim slučajevima nije uvek potrebno kroz ceo program „provlačiti" DBF datoteku, otvarati je i zatvarati po potrebi, čak i kad je ona u nekoj tesnoj vezi sa ostalim datotekama sistema. Na primer, redovan je slučaj da se kursna lista kreira kao posebna datoteka. Sa ovom datotekom će se vezivati skoro sve datoteke u sistemu, počev od raznih faktura, ulaza i izlaza itd. Stalno držati otvorenu datoteku **VALUTE** je moguće, ali nije baš pametno u slučaju velike aplikacije. Mnogo je bolje na početku programa datoteku **VALUTE** na „učitati" u jedan ili više nizova i onda u celom programu koristiti te nizove. Struktura svih datoteka bi ostala ista, jedino bi pristup elementima niza bio daleko brži nego pristup slogu datoteke. Tako bi se odgovarajući naziv valute jednostavno dobijao kao **ValNames[val_id]**, umesto da se otvara datoteka **VALUTE**, postavlja relacija **SET RELATION TO val_id INTO Valute** i traži valute->name. Pošto *Clipper 5.0* koristi disk u slučaju da mu ponestane RAM memorije, ovaj metod slobodno možete koristiti i kod veoma velikih programa.

SAJMOVI: VRT '91

NIJE RAJSKI AL' JE VRT

Od 13. do 16. novembra 1991. u Novom Sadu, u SPC „Vojvodina", a u organizaciji studija „Tega", održana je manifestacija pod nazivom „VRT '91" („vrt" je skraćenica od „vrhunska računarska tehnologija"), posvećena svemu onome što je na bilo koji način vezano za računare.

Uvek nam je predstavljalo zadovoljstvo da uđemo u prelepo zdanje sportskog centra „Vojvodina", koga mnogi bolje poznaju kao SPENS. Ali nije nam bilo ni malo drago kada smo videli da su svi izlagači smešteni u najobičniji hodnik ispred velike dvorane u kojoj su baš u vreme naše posete gostovale Nindža kornjače.

Hodnik u kome se sve održavalo bio je taman toliko velik da primi oko dvadesetak izlagača koji su došli da uprkos teškoj situaciji pokažu i dokažu da su oni još uvek tu, sa gomilom novih ideja, hardvera, softvera i usluga. Na žalost, mnogi kojima je sasvim sigurno mesto na ovakvoj priredbi nisu došli, ko zna zbog čega.

Ona slika doteranih devojaka i uglađene gospode koja sa neskrivenim samozadovoljstvom igraju tetris koju pamtimo sa ranijih sajmova, nestala je. Sada su tu simpatični ljudi sa osmehom na licu koji jedva čekaju da uputite pogled ka njihovom štandu da bi vas zasuli gomilom informacija. U tome su, kao i na većini priredbi do sada, prednjačila gospoda iz „Periharda inženjeringa" koja su nas zasipala informacijama o proizvodima koje nude, a naročito o zvezdi njihovog štanda, popularnom ink-jet štampaču „DeskJet 500" popularnog „Hewlett Packarda".

Zadovoljni onim što smo videli na „Perihardovom" štandu, zaputili smo se tamo gde smo primetili gomilu ljudi – ka štandu beogradskog „Olympa". Prvi razlog za gužvu je nova serija Epsonovih matričnih štampača koju čine modeli 570 (i njegova A3 verzija 1070) i model 870 (i odgovarajuća A3 verzija 1170). U oči prvo pada dizajn štampača koji vas ne ostavlja ravnodušnim i naravno navede vas da se raspitate za karakteristike uređaja – a tada dolazi ono pravo.

Drugi razlog gužve na štandu „Olympa" bile su atraktivne animacije urađene uz pomoć sve popularnijeg programa „3D Studio" (prikazan u „Računarima" i Animacija su izradili stručnjaci, bolje rečeno umetnici „CrossLine" studija za animaciju.

Naspram „Olympovog" štanda izlagalo je preduzeće „Info D" na čiji izložbeni prostor su nas privukli crni (!) računari. Iako su već viđena na sajmu tehnike u Beogradu, crna kućišta su i dalje veoma retka, pa zato privlače radoznalce. No, kao što reče jedan kolega, da su svi PC-ji crni, ovako bismo govorili o belim kućištima. Ovu firmu takođe treba da pohvalimo za ponudu domaćeg poslovnog softvera čije su se demo verzije „vrtele" na računarima na štandu, uveravajući nas da se radi o zaista kvalitetnim programima.

Domaćeg softvera je bilo i kod drugih izlagača, i mora se primetiti da je kvalitet primetno viši u odnosu na programe koje smo videli na ranijim manifestacijama ovog tipa. Uopšte gledano, odnos prema softveru je u mnogome promenjen. Sve je manji onih koji sa gomilama disketa idu od štanda do štanda skupljajući piratske verzije programa. Veliki broj posetilaca koji se interesovao za nabavku računara zahtevao je kompletnu uslugu koja će obuhvatiti hardver ali i softver, obuku i održavanje sistema.

Zbog nestabilnog kursa dinara, većina izlagača je u svoje cenovnike uvela tzv. bonova po kojima računaju cene, gde bonovi predstavljaju vrednost nemačke marke na „slobodnom" tržištu na dan kupovine. Valja ipak primetiti da ni teška ekonomska situacija nije opravdanje za to da niti jedan izlagač nije prodavao robu po sistemu „plati i nosi", osim neke sitnice.

Iznenadilo nas što je na sajmu bilo veoma malo laptop i notebook računara. Konkretno, videli smo samo jedan „Texas Instruments" notebook, kao i dva laptopa na štandu firme „Microsys", inače generalnog pokrovitelja sajma. Takođe, zanimljivo je da su modemske komunikacije bile potpuno zaboravljene, što veoma čudi kada se zna kakva su interesovanja kupaca ne samo u svetu nego i u nas.

Sve u svemu, vredelo je doći barem zbog nove „Epsonove" serije i HP „DeskJeta", kao i zbog same činjenice da se vrlo retko na jednom mestu skupi toliko ljudi iz oblasti računarstva. Oni koji su eventualno došli na ovu priredbu i očekivali grandioznu predstavu verovatno su otišli razočarani. Mi smo otišli zadovoljni jer smo se uverili da se i pored ogromnih teškoća sve i dalje kreće, da još nismo propustili zadnji voz za svetsku informatičku evoluciju.

Pavle Peković

DOS 5.0 ISPOD HAUBE

Pre ili kasnije, većina korisnika savlada DOS, u potpunosti ovlada svim njegovim komandama, otkrije sve trikove i „bubice"... i tu se zadovoljstvo istraživanja završava. Ili možda ne? Postoji li negde unutra čitav svet neistraženih tajni, koje čekaju na vrednog programera da ih pronađe?

Bojan Petrović

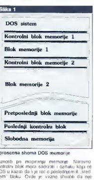
Posmatrajmo sledeći problem: razvijamo neku aplikaciju koja treba da radi pod svim verzijama DOS-a, a pritom ima potpunu kontrolu sistema i još se diže u visoku memoriju. Pretpostavimo da želimo da proverimo koliko se naš trud isplatio po memoriji. U čistom DOS-u 5.0 biće dovoljna njegova MEM komanda. Dodamo li QEMM, moraćemo upotrebiti LOADHI da bismo proverili zauzetost visoke memorije. Ako sad sve to probamo, recimo, pod DOS-om 3.30, moraćemo da nađemo neki treći program za mapiranje memorije, pošto MEM DOS-a 5.0 odbija da radi pod „dedom". Sada još poželimo da te i takve provere izvršimo iz naše aplikacije po njenom startovanju i – eto sasvim lepog problema.

Korisnici se sa ovakvim zadacima obično snalaze na gore opisani način, ali se to svodi na jedan poddirektorijum pun raznih programa tipa MEM, MI, CHKMEM, SI i drugih, koji daju podatke svaki za sebe, a ne retko se ti podaci razlikuju i međusobno. Dakle, vreme je da zađemo iza barijera komandne linije, iza leđa samog DOS-a 5.0 i pogledamo šta ima unutra.

KONTROLNI BLOK...

Ograničimo se, za početak, na osnovnu memoriju DOS-a (odnosno prvih 640 kilobajta) koja je od najvećeg interesa za većinu korisnika, a čiji je uprošćeni izgled dat na slici jedan. Kao što se vidi, DOS je memoriju podelio na blokove, pri čemu svakom bloku prethodi zaglavlje, odnosno kontrolni blok memorije („MEMORY CONTROL BLOK", „MCB"). Svaki kontrolni blok sadrži neophodne informacije koje govore DOS-u, između ostalog, koliki je blok koji sledi i ko mu je vlasnik. Pitanje vlasnika bloka memorije se može razumeti intuitivno, ali će se pokazati da je ovaj podatak od velike

Slika 1



Uprošćena shema DOS memorije

važnosti pri mapiranju memorije. Naravno, kontrolni blok mora sadržati i oznaku koja će DOS-u kazati da li je reč o poslednjem ili „srednjem" bloku. Ovde je važno shvatiti da nije neophodno označiti prvi kontrolni blok, jer je njega lako pronaći pomoću jedne druge strukture DOS-ove, „liste nad listama", o kojoj ća kasnije biti reči.

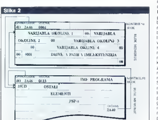
Prematome, da bismo protrčali kroz sve blokove konvencionalne memorije, treba pronaći prvi kontrolni blok, pročitati veličinu bloka koji sledi, dodati tu vrednost na veličinu samog kontrolnog bloka i dobićemo položaj sledećeg kontrolnog bloka u odnosu na početak datog. Sam blok memorije može sadržati različite stvari: program, podatke o okolini odgovarajućeg programa, delove operativnog sistema, pa čak i prazan prostor. Ovde ćemo naglasiti da DOS za svaki normalno učitan program odvaja prostor i za smeštanje podataka o okolini, što nam govori da je sa stanovišta uštede prostora pametnije sve programe („obične" ili rezidentne) učitavati što pre, po mogućnosti samo sa COMSPEC varijablom i jako kratkim PATH-om, čime smanjujemo veličinu okline („ENVIRONMENT") na najmanju vrednost.

Osmotrimo sada pažljivije strukturu kontrolnog bloka memorije prikazanu na tabeli jedan. Na ofsetu 00h nalazi se pomenuta oznaka kontrolnog bloka, dužine jedan bajt. Ako na tom mestu nađemo 4Dh (ASCII „M"), onda je reč o „srednjem", odnosno ako je tamo 5Dh (ASCII „Z") reč je poslednjem kontrolnom bloku. Treći slučaj, koji se retko pojavljuje, je kontrolni blok sa oznakom 00h, što je oznaka „nultog" kontrolnog bloka koji za dužinu bloka memorije koji sledi takođe ima upisanu vrednost 0000h. Ovakve blokove, koji se obično nalaze u grupama, najbolje je jednostavno prebrojati i zapisati njihovu ukupnu veličinu. Bilo šta drugo na ovom ofsetu znači grešku u strukturi lanca blokova memorije, pa treba voditi računa o ovoj mogućnosti (jedna od prvih lekcija dobrog programiranja) mada autor do sada nije našao na takvu situaciju.

Primetimo još da je struktura koju posmatramo jednostruko povezana lista. Kad god DOS poželi, recimo, da učita nov program, on protrči kroz celu listu da bi našao slobodno mesto, tako da neispravan podatak u nekom od kontrolnih blokova (osim na ofsetu 08h) znači siguran krah sistema.

Na ofsetu 01h kontrolnog bloka nalazi se segmentna adresa vlasnika (odnosno njegovog PSP-a, o čemu ćemo kasnije) dužine jedne reči. Ako je vrednost zapisana na tom mestu 0000h, onda blok koji sledi nije nikome dodeljen, a ako je 0008h, onda je vlasnik DOS. Za ostale vrednosti je nemoguće proveriti da li su neispravne (zapravo je moguće, ali to prevazilazi ambicije ovog teksta). Na ofsetu 03h, dužine jedne reči, nalazi se veličina bloka memorije koji sledi, data u 16-bajtnim paragrafima (što znači da će se pretvaranje ove veličine u broj bajtova svesti na neki šift levo). Sledeća tri bajta se ne koriste i konačno na ofsetu 08h (od DOS-a 4.0 naviše) nalazi se 8 bajtova imena programa datog u ASCII formatu (ako je bilo šta drugo, imena neće biti). Ako je ime kraće od osam slova, završava se sa nulom, ali će u mnogim slučajevima umesto imena na tom mestu biti neupotrebljive vrednosti. Uostalom, u tih osam bajtova može da stane eventualno ime programa, ali ne i njegova ekstenzija, tako da ćemo tačan naziv programa tražiti na drugi način.

Slika 2



Veze između imena programa i okoline

...I BLOKOVI MEMORIJE

I tako smo stigli do samog bloka memorije čiji smo početak saznali na gore opisani način. Problem koji logično sledi je određivanje sadržaja bloka, odnosno tipa podataka koji su u njemu smešteni. Pre svega, ako je vlasnik 0000h, blok je slobodan za dodeljivanje i često ćemo ovakve strukture nalaziti usred lanca

Struktura kontrolnog bloka memorije

OFSET	VELIČINA	OPIS SADRŽAJA
00 h	BAJT	Tip kontrolnog bloka
01 h	REČ	Segment PSP-a vlasnika
03 h	REČ	Veličina bloka u paragrafima
05 h	3 BAJTA	Ne koristi se
08 h	8 BAJTOVA	Ime programa (DOS 4 0+)

Struktura PSP

OFSET	VELIČINA	OPIS SADRŽAJA
00 h	2 BAJTA	INT 20 h ili INT 27 h
02 h	REČ	Segment prvog bajta posle bloka
04 h	BAJT	Ne koristi se
05 h	5 BAJTOVA	FAR poziv INT 21 h
0A h	DVE REČI	Završetak zamenjenog INT 22 h
0E h	DVE REČI	Završetak zamenjenog INT 23 h
12 h	DVE REČI	Završetak zamenjenog INT 24 h
16 h	REČ	Segment očevog PSP-a
18 h	36 BAJTOVA	Rezervisano za DOS
3C h	4 BAJTA	Ne koristi se
40 h	2 BAJTA	Verzija DOS-a koja se prijavljuje
42 h	14 BAJTOVA	Ne koristi se
50 h	3 BAJTA	INT 21 h RETF instrukcija
53 h	9 BAJTOVA	Ne koristi se
5C h	16 BAJTOVA	Prvi pretpostavljeni FCB
6C h	16 BAJTOVA	Drugi pretpostavljeni FCB
7C h	4 BAJTA	Ne koristi se
80 h	128 BAJTOVA	Komandna linija

Lista nad listama

OFSET	VELIČINA	OPIS SADRŽAJA
– 8	DVE REČI	Pokazivač na trenutni disk bafer
– 4	REČ	Pokazivač na nepročitani deo ulaza sa CON uređaja 0000 h ako nema nepročitanog dela
– 2	REČ	Segment prvog kontrolnog bloka
00 h	DVE REČI	Pokazivač na prvi blok Drive parametara
04 h	DVE REČI	Pokazivač na prvu tabelu sistemskih fajlova
08 h	DVE REČI	Pokazivač na zaglavlje aktivnog CLOCK$ uređaja
0C h	DVE REČI	Pokazivač na zaglavlje aktivnog CON uređaja
10 h	REČ	Maksimalan broj bajtova bilo kog BLOK uređaja
12 h	DVE REČI	Pokazivač na zapis informacija o disk baferima
16 h	DVE REČI	Pokazivač na niz trenutnih strukture direktorijuma
1A h	DVE REČI	Pokazivač na sistemske FCB tabele
1E h	REČ	Broj zaštićenih FCB
20 h	BAJT	Broj instaliranih BLOK uređaja
21 h	BAJT	Broj dostupnih DRIVE slova
22 h	18 BAJTOVA	Zaglavlje NUL uređaja
34 h	11 BAJTOVA	Interne varijable
3F h	DVE REČI	X i Y iz komande BUFFERS x y
43 h	BAJT	Boot drive (1 A)
44 h	BAJT	Nepoznato
45 h	REČ	Veličina EXTENDED memorije

blokova. Ako je veličina slobodnog bloka dovoljno velika, DOS će u njega smestiti okolinu nekog programa koga ćemo ponekad naći tek nekoliko blokova dalje. Ako se u datom bloku memorije nalazi program (ili njegova okolina), onda ćemo na adresi vlasnika pronaći zaglavlje koje ćemo dalje označavati sa PSP („PROGRAM SEGMENT PREFIX"). Zato ćemo od sada prvo pronalaziti adresu mogućeg PSP-a i to koristiti da proverimo da li je stvarno reč o programu, njegovoj okolini ili, recimo, sistemskom delu, kada PSP-a nema.

Struktura PSP-a je data na tabeli dva. Na ofsetu 00h nalaze se dva bajta pomoću kojih ćemo znati da li je stvarno reč o programu ili ne. Naime, u ta dva bajta DOS smešta 20CDh ili 27CDh (mada retko), što su asemblerske instrukcije INT 20h odnosno INT 27h. Ponekad se rad programa okončava skokom na ovu adresu, ali je u opštem slučaju sigurnije koristiti funkciju 4Ch prekida 21h. Sledeći podatak od interesa se nalazi na ofsetu 16h, što je segment „očevog" PSP-a. Ovaj podatak nam omogućava da pronađemo prvi PSP, tako što ćemo u svakom PSP-u čitati segment prethodnog i vraćati se unazad. Prvi PSP u lancu će biti onaj COMMAND.COM-a (4DOS.COM-a ili bilo kog drugog komandnog procesora), gde će na ofsetu 16h stajati njegova sopstvena adresa. Ofset 2Ch sadrži segmentnu adresu okoline programa, što će nam pomoći da ustanovimo kad smo naleteli na okolinu programa.

Na ofsetu 80h svakog PSP-a nalaziće se kompletna komandna linija kojom smo startovali program, sem u slučaju COMMAND.COM-a (nekog drugog komandnog procesora ili sistemskog dela). Odatle sledi da bismo ime programa mogli naći ispitivanjem komandne linije ako je program u pitanju, odnosno iz kontrolnog bloka memorije ako je komandni procesor u pitanju. Da bismo znali kada je reč o komandnom procesoru (ili njegovoj okolini) proveričemo šta je upisano na ofsetu 16h njegovog PSP-a, na gore opisani način. Međutim, u nekim slučajevima se događa da podaci sa ofseta 6Ch PSP-a pređu preko komandne linije, tako da ćemo za određivanje imena programa koristiti podatke dobijene iz okoline. Interesantno je pomenuti i ofset 40h, koji u dužini od dva bajta sadrži verziju DOS-a koja će biti prijavljena ako se zatraži funkcijom 30h prekida 21h.

Sada možemo razlikovati program i njegovu okolinu. Ako je program, adresa vlasnika će biti prvi bajt posle kontrolnog bloka i tu će biti ili INT 20h ili INT 27h (lepše rečeno, ako je adresa vlasnika sledeći paragraf...). Ako je okolina, adresa vlasnika neće biti bajt posle kontrolnog bloka, ali će na tom mestu opet biti pomenuta dva prekida. Okolina sadrži neke varijable sistema koje smo mogli definisati u AUTOEXEC-u (ili ih je sistem sam odredio), recimo PATH, PROMPT, COMSPEC, ali i puno ime programa čija je okolina u pitanju, uključujući disk i ceo put do programa. Svaka varijabla je zapisana u ASCIZ obliku, što je običan ASCII niz koji se završava nulom. Na kraju poslednje varijable (i njene nule) nalazi se još jedna nula koja označava kraj ispisa varijabli, a zatim reč koja sadrži broj stringova koji slede. Najčešće je broj stringova 0001h, ali pošto autor još nije video drugu situaciju, ova vrednost se sa sigurnošću može upotrebiti da bi se našao početak imena programa o čijoj se okolini radi. Ovog puta nema bojazni da će na neki način ime programa biti uništeno, pa ćemo ovaj podatak iskoristiti da pronađemo ime i programa i okoline. Ove naizgled komplikovane veze prikazane su na slici dva.

Ako, međutim, pronađemo naše INT 20h ili INT 27h na njihovom mestu, a se pokaže da ostali uslovi nisu zadovoljeni, onda pretpostavimo da je reč o bloku sa podacima (tip „DATA"), dok ako nista od prethodnog ne uspe, onda pretpostavimo da je reč o sistemu, u kom slučaju ne tražimo ime. Do sada je autor našao samo na jedan slučaj kada se pojavio DATA blok, tako da nije jasno šta on stvarno predstavlja, i može se pretpostaviti da je taj blok program naknadno dodelio sam sebi, kako bi u njega smestio neke svoje informacije. Druga ograničenja na koja se nailazi se tiču verzije DOS-a. Recimo, u verzijama starijim od DOS-a 4.0, kontrolni blok ne sadrži ime programa, tako da tada ime komandnog procesora ne možemo saznati.

LISTA NAD LISTAMA

Struktura o kojoj ćemo samo ukratko je do sada bila nedokumentovana i važila je za internu stvar DOS-a. Ipak, među programerima je bila poznata i to pod mnogim imenima: „SYSVARS", „IN-DOS", „LIST of LISTS" i druge. Reč je o strukturi koja sadrži pravu gomilu pokazivača na neke druge osnovne strukture DOS-a, sistemske tabele i razne druge informacije koji se koriste interno. Jedini način da se dopre do liste nad listama je DOS-ova funkcija 52h, koja vraća u paru ES:BX pointer na početak ove liste. Kao i ranije, pomoću ofset adresa naći ćemo sve što nas interesuje. Recimo, na ofsetu -2 nalazi se segment prvog kontrolnog bloka memorije. Pre nego što se saznalo za ovaj „link", većina programa je tražila prvi kontrolni blok „backtracking"-om, odnosno praćenjem unazad adresa PSP-ova, što je već opisano, dok je sada taj posao daleko olakšan. Na ofsetu 22h nalazi se zaglavlje NUL uređaja koji je uvek prvi u listi instaliranih uređaja sistema. Struktura liste je data u tabeli tri. Treba napomenuti da se lista menjala sa verzijom DOS-a, tako da su samo elementi liste zaključno sa ofsetom 0Ch zajednički za sve

Neke funkcije MULTIPLEX prekida

FUNKCIJA	ODGOVOR	SVRHA
AX – 0001 h	AL 00 h nije instaliran moguca instalacija AL – 01 h nije instaliran ne instalirati AL – FF h instaliran	PRINT COM OK provera instalacije
AX 0600 h	AL 00 h nije instaliran moguca instalacija AL – 01 h nije instaliran ne instalirati AL – FF h instaliran	ASSIGN COM OK provera instalacije
AX = 0800 h	AL – 00 h nije instaliran moguća instalacija AL – 01 h nije instaliran ne instalirati AL – FF h instaliran	DRIVER SYS provera instalacije
AX – 1000 h	AL – 00 h nije instaliran moguca instalacija AL = 01 h nije instaliran ne instalirati AL = FF h instaliran	SHARE OK instalirati provera instalacije
AX = 1400 h	AL = 00 h nije instaliran moguća instalacija AL = 01 h nije instaliran ne instalirati AL = FF h instaliran	NLSFUNC COM provera instalacije
AX = 1600 h	AL = 00 h nije instaliran AL = 80 h nije instaliran AL = 01 h Win 2 X instaliran AL = FF h Win 2 X instaliran inace AL = visi broj verzije Windows a AH = niži broj verzije Windows-a	MS WINDOWS (Enhanced mod) provera instalacija
AX = 1A00 h	AL FF h instaliran	DOS 4+ ANSI SYS provera instalacije
AX = 4300 h	AL = 80 h instaliran	XMS driver provera instalacija
AX = 4680 h	AX = 0000 h instaliran	MS WINDOWS 3 X (Real ili Standard) provera instalacija
AX = 4800 h	AL = 00 h nije instaliran	DOS 5 0 DOSKEY provera instalacije
AX – AD80 h	AL = FF h instaliran BX = broj verzije ES DI pokazivac na interno podatke	KEYB COM provera instalacija

IMPLEMENTACIJA

Obicno se smatra da Turbo Pascal nije najsrecniji izbor za sistemsko programiranje ali ne možemo reci da je to uvek tačno Ovaj put cemo namerno pokušali da nas zadatak u celini realizujemo koristeci samo legalne metode Turbo Pascala 5 5 ne samo da bismo pokazali koliko je to jednostavno već i zato što autor mnogo voli paskal

Pre svega treba nam adresa liste nad listama U tom cilju iskoristicemo proceduru INTR i tip REGISTERS koje cemo naci u UNIT-u DOS"

USES DOS

VAR r REGISTERS

BEGIN

AH S52,

INTR (S21 r),

END

Naravno mogli smo umesto procedura INTR upotrebili MSDOS proceduru koja podrazumeva da je reč o prekidu 21h Na ovaj nacin u paru ES BX dobili smo adresu liste nad listama pa nam ostaje da pronademo prvi kontrolni blok cij segment nalazimo na ofsetu -2

segm konlr – MEMW(r ES r BX-2)

Vec smo rekli da je lanac kontrolnih blokova jednostruko povazana lista sto nas odmah navodi na ideju da iskoristimo pokazivace Medutim necemo ni pokusavali da koristeci NEW dodeljujemo memoriju pokazivačima jer cemo samo privremeno pristupali nekim delovima već dodeljene memorije Sa druge strane nećemo koristiti ni DISPOSE da se oslobađamo memorije dodeljene pokazivacima jer je i nismo dodelili Ukoliko konsticemo pokazivače samo da pristupimo i eventualno čitamo podatka od interesa ali ništa više Naravno moramo definisati tip vrijable koju camo koristil za citanje podataka o kontrolnim blokovima i opet nam se odgovor sam nametnuo Koristicemo tip zapisa jer on verno odslikava strukture kojima zelimo da pristupimo (listing 3)

Pošlo je velicina data u 16 bajtnim paragrafima preracunacemo je u bajtove u LONGINT tipu korištenjem SHL pomeraja

velicina_b LONGINT(k_blok\ velicina SHL 4),

Ostaje nam još da pronademo PSP i okolinu što cemo uraditi na potpuno isti način kao za kontrolni blok (listing 4)

Sam proces pretraživanja bloka memorije i otkrivanja njegovog tipa zbog obimnosti ostavljamo citaocu Pošto smo završili posao sa datim kontrolnim blokom prelazimo na sledeći jednostavnim sabiranjem njegovog segmenta i veličine njega samog pri čemu vodimo racuna da je veličina kontrolnog bloka jedan paragraf

segm_kontr –segm konlr –k_blok\ velicina–1

I to bi bilo to Ako se pažljivo prate objasnjenja data u tekstu čitalac ce moci da napravi zgodan program za mapiranje memorije „po svojoj meri"

Listing 1

```
PROGRAM postoji_li_qemm,
USES DOS,
VAR r  REGISTERS;
BEGIN
  r AX =$D201,
  r BX =$5145, { ASCII "QE" }
  r CX =$4d4d, { ASCII "MM" }
  r DX:=$3432, { ASCII "42" }
  INTR ($2F,r),
  IF (r.BX=$4F48) THEN WRITELN ( QEMM instaliran )
END.
```

Da li je instaliran QEMM

Listing 2

```
PROGRAM segment_prvog;
USES DOS,
VAR r: REGISTERS;
BEGIN
  r AX:=$0201;
  r BX =$4849; { ASCII "HI" }
  r CX:=$5241; { ASCII "RA" }
  r DX =$4D30; { ASCII "MO" }
  INTR ($2F,r);
  IF (r BX=$4F48) THEN WRITELN ( 'Prvi blok je na: ',
                                  (LONGINT (r CX)) SHL 4)
END
```

Pronadi kontrolni blok u visokoj memoriji

Listing 3

```
TYPE kontrolni_blok=
RECORD
   oznaka        BYTE,
   vlasnik     : WORD;
   velicina    : WORD,
   nebitno     : ARRAY [1..3] of BYTE;
   ime_programa: ARRAY [1..8] of CHAR;
 END;
 VAR k_blok ^kontrolni_blok;
 BEGIN
   k_blok:= PTR(segm_kontr,0);
 END;
```

Pronadi kontrolni blok

Listing 4

```
TYPE programsegmentprefix=
RECORD
  psp_oznaka  WORD,
  nebitno     :ARRAY [1..10] of WORD,
  otac        :WORD,
  nebitno_1   :ARRAY [1..10] of WORD,
  okolina     :WORD;
END;
VAR psp. ^programsegmentprefix,
BEGIN
  psp.= PTR(k_blok^.vlasnik,0);
END
```

Pronadi PSP

HMA, SETVER, QEMM...

Jedna od „revolucionarnih" novina DOS-a 5.0 je njegova mogućnost da rezidentne programe smesti u visoku memoriju (po Microsoftu je to „UMB" odnosno „UPPER MEMORY BLOK"). U duhu prethodnog teksta pokušaćemo da pronađemo i njih. Ovog puta podatak koji nam je potreban, segment prvog kontrolnog bloka, neće biti u listi nad listama ali ćemo do njega doći posredno preko jednog drugog podatka koji na prvi (a i drugi) pogled nema neke veze sa visokom memorijom.

Na ofsetu 12h liste nad listama (što nalazimo na ranije opisan način) nalazi se pokazivač na zapis informacija o baferima diska („DISK BUFFER INFO RECORD"). Očigledno je da većina informacija iz ovog zapisa ima eventualno posredne veze sa visokom memorijom jer se baferi mogu (a ne moraju) smestiti u nju. Recimo, na ofsetu 0Ch se nalazi bajt koji nam govori da li smo bafere podigli u visoku memoriju ili su u konvencionalnoj. Ipak, na ofsetu 1Fh nalazi se segment prvog kontrolnog bloka u visokoj memoriji. Ako se na tom mestu nađe FFFFh, onda DOS poseduje blokove samo u konvencionalnoj memoriji, a u bilo kom drugom slučaju postupamo kao i ranije. Lanac kontrolnih blokova u visokoj memoriji je organizovan na potpuno isti način kao u konvencionalnoj, uključujući i oznake kontrolnih blokova. Tako da se sve što smo rekli za blokove u konvencionalnoj memoriji odnosi i na one u visokoj. Prvi kontrolni blok visoke memorije se obično nalazi na 9FFFh.

Još jedna novost DOS-a 5.0 je SETVER, komanda koja „vara" programe prijavljujući im verziju DOS-a koju oni zahtevaju. Naime, veliki broj programa pisanih pre pojave DOS-a 5.0 zahteva određenu verziju DOS-a, i odbija da radi pod drugim. Ovom komandom možemo za svaki takav program, ali i bilo koji drugi, zadati verziju DOS-a koja će biti prijavljena kada to program bude zahtevao. Na koji način proveriti da li vas SETVER pokušava prevariti? Prvo rešenje se svodi na činjenicu da je SETVER rezidentan, pa bi ga možda mogli naći mapiranjem memorije i traženjem programa sa imenom „SETVER.EXE" po blokovima memorije.

Međutim, postoji i elegantnije rešenje, zbog čega ćemo se poslužiti više puta pomenutim prekidom 21h. Većina programa proverava verziju DOS-a pozivom prekida 21h, funkcija 30h, pri čemu DOS (bilo koje verzije) u AL vraća viši, u AH niži broj verzije, a u paru BL:CX 24-bitni serijski broj. Kod DOS-a 5.0 možemo pri pozivu u AL staviti 01h, što će nam u BH vratiti 10h ako je DOS u visokoj memoriji. Nova funkcija koju je uveo najmlađi DOS je 33h, podfunkcija 06h (odnosno AX=3306h), kada DOS 5.0 vraća u BL viši broj, u BH niži broj verzije, u DL reviziju i u DH setovan bit 4 ako je DOS u visokoj memoriji. Prema tome, jednostavan način da proverimo da li SETVER pokušava da nas prevari je da uporedimo veličine koje nam vrate ove funkcije.

Sada bi bilo nepravedno zaobići QEMM. Ogroman broj korisnika koji se odlučio da pređe na DOS 5.0 je ipak zadržao QEMM, jer ovaj još uvek pruža više komotnosti nego DOS. Očigledno je da, ako želimo da zaokružimo temu, moramo da pronađemo programe koje smo podigli u visoku memoriju pomoću QEMM-ova LOADHI komande. Radi toga upoznaćemo se sa još jednom interesantnom nepoznanicom DOS-a, prekidom zvanim MULTIPLEX.

MULTIPLEX je prekid 2Fh. U poslednje vreme, pošto su skoro sva slobodna mesta prekida 21h (DOS funkcije) iskorišćena, ovaj prekid se sve više koristi za opštenje sa nekim specifičnim aplikacijama. MULTIPLEX-u nije potrebna podrška DOS-a da bi se koristio u aplikacijama, niti se o njemu DOS uopšte i brine, već o njemu brinu sami korisnici. Kao i kod prekida 21h, kod ovog prekida imamo funkcije čiji je redni broj smešten u AH, što govori kom programu vaš zahtev treba da se uputi. Ako želite da se vaša aplikacija posluži prekidom 2Fh, morate pre svega proveriti da li je neki drugi program već zauzeo željenu funkciju. Ako nije, na lanac postojećih hendlera dodajete svoj, i time ste rezervisali određenu funkciju za svoje potrebe. Ipak, uobičajeno je da se tuđa mesta ne zauzimaju, već se isključivo traže slobodna.

Spisak aplikacija koje koriste usluge MULTIPLEX-a je ogroman, ali među njima vredi potražiti neka poznata imena. Recimo, DOS-ovi DRIVER.SYS, NLSFUNC, DISPLAY, GRAFTABL, APPEND, PRINT, ASSIGN, GRAPHICS pa i novi DOSKEY su zauzeli svako svoju funkciju (neki i po nekoliko njih). Veliki broj funkcija služi za proveru statusa određene aplikacije, ali i za proveru da li je aplikacija uopšte instalirana. Međusobno opštenje se vrši slanjem tačno određenih vrednosti smeštenih u registre, i proveravanjem istih.

Kompanija „Quarterdeck", čija su dela QEMM, QRAM i DesqView, ima jako lep „potpis" za svaki od svojih proizvoda. Njihova funkcija MULTIPLEX-a je D201h (odnosno AX=$D201), koja se koristi za proveru da li je neka njihova aplikacija prisutna, ali i da bi se saznale još neke interesantne činjenice. U našem slučaju, postupićemo kao na listingu 1.

Kao odgovor, ako je QEMM (QRAM se odaziva na isti način) instaliran, u BX će se vratiti 4F4Bh (ASCII „OK"). Kad se uverimo da je QEMM prisutan, trebalo bi pronaći gde je prvi kontrolni blok u visokoj memoriji QEMM-a. Opet MULTIPLEX (listing2.pas).

Opet, ako postoji lanac blokova u visokoj memoriji, u BX će biti 4F4Bh („OK") ali ćemo u tom slučaju u CX dobiti segment prvog kontrolnog bloka, a u DX segment na kom se nalazi sam QEMM. Lanac blokova QEMM-a u visokoj memoriji je opet iste strukture kao onaj u konvencionalnoj. Tako da sad možemo napraviti program koji će biti svestan prisustva QEMM-a i DOS-a 5.0 u bilo kojoj kombinaciji. Na tabeli celini mogu se videti informacije o nekim funkcijama MULTIPLEX-a.

UZMITE SA SEZAMA

Program koji demonstrira sve što je rečeno u ovom tekstu može se pronaći na Sezamu pod nazivom BMEM (\IBMPC\UTILITES\BMEM.ZIP). Njegova osnovna funkcija je mapiranje memorije DOS-a 5.0, bez obzira da li je u konvencionalnoj ili visokoj memoriji; pronalazi programe u DOS 5.0 visokoj memoriji, QEMM-ovoj visokoj memoriji, QRAM-ovoj visokoj memoriji; svestan je prisustva 4DOS-a, Quarterdeckovog MANIFEST-a i VIDRAM-a. Ako pri startovanju zadamo parametar „FREE", BMEM prijavljuje slobodnu konvencionalnu, XMS i EMS memoriju, kao i još neke interne podatke sistema. Konačno, ako se zada parametar „BOOT", u varijabli ERRORLEVEL se vraća drajv sa koga smo podigli sistem (ERRORLEVEL=1 znači „A:"), što se može zgodno iskoristiti u nekom BATCH programu ili AUTOEXEC-u. Naravno, uz malo proučavanja svaki čitalac će moći da napravi i svoj program od BMEM-a po svojoj meri.

ŠTA IMA NOVO

LASER TEŠKE KATEGORIJE

Mannesmann Tally je predstavio novi laserski štampač brzine 10 strana u minutu, što predstavlja dvostruko više od industrijskog standarda. MT911 koristi heavy duty mehanizam i MT-ov JADE kontroler koji obezbeđuje veoma brzo procesiranje i radni vek od 600.000 kopija. Kontroler raspolaže sa 100 rezidentnih fontova, uključujući i sve HP Series II fontove, kao i fontova iz S1, S2, T i Z kartridža, a za one kojima ni to nije dovoljno su predviđena dva HP kompatibilna slota. Štampač je serijski opremljen magacinom koji prima 200 listove papira (na raspolaganju je i opcija od 500 listova), face down izlazni držač za 200 listove i standardni face up izlaz. Papiri vanserijske težine (do 120 gr) se mogu ubacivati i ručno.

MT 911 je standardno opremljen sa 512 KB RAM-a a na raspolaganju su i proširenja od 1, 2 i 4 MB. Tip emulacije je HP LaserJet Series II, a sa računarom se povezuje preko Centronics ili RS 232 interfejsa.

Cena: $1599 za štampač sa standardnom opremom, $275 za dodatni držač papira za 500 listova, $99 za proširenje od 1MB, $209 za 2 MB i 399 za 4 MB.

Kontakt adresa: Mannesmann Tally Ltd., Molly Millar's Lane, Wokingham, Berkshire RG11 2QT, U.K., tel. 44 734 788711, fax 44 734 791401

BIBLIOTEKAR ZA IKONE

Kolekcije ikona, kao i svaka druga kolekcija, može lako da se uveća toliko da snalaženje u njoj postane nemoguće. Korisnicima koji imaju takve probleme je namenjen *Icon Manager 1.2* firme *Impact Software* – program omogućava pakovanje ikona u vidu individualnih fajlova ili u vidu biblioteka u formatu koji Program Manager može da pročita, a omogućeno je i držanje kompresovanih arhiva. Biblioteke se prikazuju u višestrukim prozorima nazvanim *listovi* uz mogućnost direktnog učitavanja ikona iz direktorijuma .EXE fajlova ili DLL fajlova, a istovremeno se može obrađivati 30 listova.

Mada pravo editovanje ikona nije moguće, program dozvoljava neke sitnije intervencije kao što su formiranje transparentne podloge ili invertovanje atributa ikone. Pored toga, na raspolaganju je i poseban alat kojim je moguće bilo koju oblast dimenzija 32 x 32 piksela pretvoriti u ikonu, a uz program se isporučuje i kolekcija od 250 ikona.

Icon Manager je shareware program, može se direktno download-ovati sa PC MagNet-a (UtilForum), a cena registracije iznosi $24.95.

Kontakt adresa: Impact Software, 12140 Central Ave., #133, Chino, CA 91710, tel. 714 590-8522

WINDOWS MENADŽER

Norton Desktop for Windows je moćan program velike funkcionalnosti, ali je za mnoge korisnike preskup – takvima je namenjen *Selvation* firme *Vitesse Inc.* po ceni od $180. Program predstavlja *Windows* školjku koja zamenjuje i *ProgMan* i *FileMan* i umesto njih postavlja sopstveni interfejs. Korisnik može da izabere startni direktorijum i da podesi dodatne parametre pri svakom pokretanju programa. Ceo sistem je baziran na tzv. *radio button* stranama, čiji izgled zavisi od rezolucije, a pored biranja programa su omogućene i druge pomoćne funkcije.

Kontakt adresa: Vitesse Inc., P.O. Box 929, La Puente, CA 91747-0929, tel. 800-777-7344

PO ISTINITOM DOGAĐAJU

Prelazeći sa velikih računara, programeri su, zajedno sa klasičnim programskim jezicima, na PC preneli i tradicionalne metode programiranja, ali su brzo počeli da se udaljavju od njih – posle strukturnog i objektnog, na scenu stupa i programiranje zasnovano na događajima (*event driven*).

Dejan Jelović

Tradicionalne metode u početku nisu predstavljale nerešiv problem: aplikacije su bile relativno male i nije bilo teško snaći se u njima. Međutim, čim je dostignut određeni stepen složenosti programa, problemi su počeli da se pojavljuju svuda. Uzmimo za primer neku jednostavnu bazu podataka. Ako imamo formu za unos iz koje se ne može izaći sve dok se u jedno polje ne unese neka od korektnih šifri za neki artikal, korisnik koji je zaboravio šifre upada u mrtvu petlju. Da bi pogledao šifre, mora izaći iz forme za unos; da bi izašao iz forme za unos, mora da zna šifru. Izlaz je jedino u pritisku na crveno dugme.

Ovaj problem je zaista lako otkloniti: treba samo ugraditi opciju privremenog izlaska iz forme za unos i pregleda baze podataka koje sadrži šifre. Međutim, šta da se radi ako program treba da barata sa više desetina baza? Ne samo da bi program dosta „otežao" od ovih silnih dopuna, nego bi i trebalo pohvatati sve relacije između raznih baza podataka, što nije nimalo naivan posao.

ZASNOVANO NA DOGAĐAJU

Za rešenje nije trebalo ići daleko – XEROX PARC je za potrebe grafičkih korisničkih interfejsa razvio novi pristup pisanju programa, tzv. *event driven* (ED) tehnike programiranja. (*Event driven* bi se moglo prevesti kao „na događajima zasnovano (programiranje)", gde događaj može biti pomeranje miša, pritisnuta dirka, znak na COM portu... bilo šta na šta bi računar trebalo da reaguje. Kako je ovakav prevod relativno nespretan za stalno korišćenje, mi ćemo se zadržati na engleskom nazivu.)

Prvi program koji je tipičnog PC programera uveo u ED svet bio je *MS Windows*. Svako ko je želeo piše za ovaj operativni sistem morao je da se pozabavi ED programiranjem. *Windows* međutim, nije bio prihvaćen kao što je to Microsoft planirao, tako da je ceo projekat privremeno pao u senku (sve do verzije 3.0, koja je napravila pravu malu revoluciju).

Nekoliko godina kasnije, na scenu stupa Borland koji čini pionirski korak: uz svoj objektni paskal svakom korisniku daje i *Turbo Vision* – ED biblioteku za pravljenje grafičkih korisničkih interfejsa. Iako je masa ljudi po inerciji nastavila da piše programe kao što su to radili i do sada, ED programiranje se definitivno ustalilo i trend u tom smeru je očigledan.

Iako bi mnogi prodavci programske opreme želeli da vas ubede da je ED programiranje „nešto sasvim novo", promena je mnogo manje radikalna nego što to izgleda. Program je jednostavno podeljen na deo koji ćemo zvati *handler*, zadužen za „hvatanje" događaja, i na mnoštvo *procedura* kojima *handler* prosleđuje događaje koje one treba da obrade. Procedura predstavlja skup funkcije koje *handler* poziva kada budući da se određeni događaj odnosi na njih. U integrisanom TC++ okruženju procedure predstavljaju editor, meni, dijalozi... a *handler* bi predstavljala funkcija koja čeka pritiske na dirku ili klik mišem. Srž ED programiranja je u sledećem: izlaskom iz jedne procedure podaci u njoj ostaju netaknuti, tako da ponovnim ulaskom u nju možete nastaviti **tačno** tamo gde ste stali. Recimo, ako u TC++ uđete iz editora u meni pritiskom na F10, kada iz tog menija izađete sa ESC vraćate se na potpuno isto mesto u editoru iz koga ste krenuli! Jednostavno i moćno.

ED OOP

Programske biblioteka za razvoj ED programa, o kojoj je ovde reč, nastala je prilikom pravljenja grafičkog interfejsa koji bi bio prenosiv na velikom broju platformi. Iako je ovde C (upravo zbog svoje prenosivosti) delovao kao logičan izbor, odluka je ipak pala na C++. Iako bi se kao argument u prilog tome ovde mogli pisati hvalospevi o objektnim (OO) jezicima, ograničimo se za trenutak na konkretne prednosti C++-a nad C-om.

- Već je rečeno da proceduru predstavlja skup funkcija. Umesto da se u C-u maltretiramo sa pointerima na sve ove funkcije, možemo ih je staviti u klasu i koristiti jedan pointer za referenciranje funkcija unutar cele klase.
- Polimorfizam operatora (engl. *operator overloading*) predstavlja mnogo elegantnije rešenje od funkcija sa promenljivim brojem parametara. Kada *handler*-u treba prijaviti koje sve dirke služe za poziv neke procedure, mnogo je lakše napisati

handler + menu_proc – F10 – ALT_M

nego

addkeys (&handler, _pkk_ &menu_proc F10, ALT_M)

- Brigu o podacima neke procedure preuzimaju konstruktori i destruktori, tako da je korisnik klase oslobođen potrebe da ih ručno inicijalizuje i da na kraju „počisti".
- Virtualne funkcije predstavljaju mnogo bolje rešenje od pointera na funkcije (mada se svode na isto) ili od **switch** naredbe. (Svako ko vidi **switch** naredbu na četiri strane iz *MS Windows SKD*-a nagio bi, budući da se pozabavi tim objektnim programiranjem".)

Ovakav pristup ima i svoje mane – najveća od njih je neprenosivost. Ne samo da ne postoji ANSI standard za C++, nego i ne postoje C++ prevodioci za sve rasprostranjene sisteme. Iako se C++ programi mogu prevesti na C pomoću UNIX-ovog CFRONT-a, za programsku biblioteku nema spasa – ima je svrhe koristiti samo na sistemu koji podržava C++.

BIBLIOTEKA DRIVER

Biblioteka se sastoji od strukture **event** koja služi za opis događaja, klase **proc** i klase **handler**. Ideja je u sledećem: nove procedure se definišu kao naslednici klase **proc**, zamenjujući njene virtualne funkcije novim. Tako definisane procedure prijavljujemo instanci klase **handler** izrazom tipa

```
handler    + menu – F10
           + editor – ALT_1
           ...
           + project – ALT_P;
```

Sve što sada treba uraditi jeste pokrenuti sam *handler* komandom

handler.run (editor id);

i *Voila!* – imamo ED program.

Da bi se koristila ova biblioteka, treba u tekst programa uključiti *include* datoteku DRIVER.HPP. (Iako su C++ *include* datoteke koje dolaze uz *Turbo C++* obeležene sa .H, ja sam zadržao staru naviku da ih obeležavam sa .HPP. Tako je lakše snaći se u šumi C i C++ datoteka na disku.) Prva deklaracija odnosi se na strukturu **event**.

Listing 1 DRIVER.CPP

```
#include "driver.hpp"
#include <conio.h>

// ***** Implementacija apstraktne klase proc *****

proc::proc (int ret)

// Konstruktor klase proc
// Parametar ret se direktno prenosi u RETURNABLE.

{
 RETURNABLE = ret;
}

void proc::init (void) {}

// Funkcija poziva handler svaki put kada
// se sa druge procedure predje na ovu

void proc::done (void) {}

// Funkcija se poziva svaki put kada handler dobije
// signal da treba da predje na novu proceduru

#pragma argsused
void proc::run (event &ev) {}

// Poziva handler kada je neato desi
// Parametar ev predstavlja dogadjaj zbog koga je funkcija pozvana


// ***** Implementacija klase handler *****

event handler::getevent (void)

// ova funkcija vraca pritisnute dirke ukljucujuci i dirke sa "tive"
// tastature, osim ako ima nesto na eventStack-u, u kom slucaju evente
// skida odatle

{
 if (eventStackP)
      return (eventStack [--eventStackP]);

 event ev;
 int key;

 ev.type = KEYIN;
 ev.i1 = ((key = getch ())) ? key : (0x100 + getch ()));
 return ev;
}

handler::handler (void)

// Konstruktor klase handler
// Inicijalizuje tabele

{
 exitflag = 1;
 koliko = 0;
 eventStackP = 0;
 for (int i = 0; i <= 0xiff; i++)
      keyvec [i] = -1;
}

handler& handler::operator + (proc& pr)

// handler + proc registruje objekat proc kod handlera
// a objektu proc prosledjuje njegov id i adresu handlera

{
 prvec [koliko] = &pr;
 pr.id = koliko++;
 pr.handlerp = this;
 return *this;
}

handler& handler::operator = (int key)
// postavlja "hotkey" za zadnju registrovanu proceduru

{
 keyvec [key] = koliko - 1;
 return *this;
}

int handler::event2proc (event& ev)

// U zavisnosti od dogadjaja (event) ova funkcija bira
// proceduru koja ce biti pozvana

{
 int id;

 if (ev.type == KEYIN) {
      if ((id = keyvec [ev.i1]) == -1)
           return current;
      else {
           ev.type = HOTKEY;
           return id;
      }
 }
 if (ev.type == CALL)
      return ev.i1;
}

void handler::insertEvent (event& ev)

// Ubacuje event na stek - citace ih kasnije funkcija getevent

{
 eventStack [eventStackP++] = ev;
}

void handler::run (int pr)

// Na osnovu dogadjaja poziva proceduru sve dok je
// exitflag jednak 1

{
 int lastCalled;
 lastCalled = current = pr;
 event ev;
 proc *prlast, *procurr;

 (prvec [pr]) -> returnTo = pr;
 procurr -> init ();
 do {
      current = event2proc (ev = getevent ());
      if (current != lastCalled) {
           (prlast = prvec[lastCalled]) -> done ();
           procurr = prvec [current];
           if (prlast -> RETURNABLE)
                procurr -> returnTo = lastCalled;
           else
                procurr -> returnTo = prlast -> returnTo;
           procurr -> init ();
      }
      procurr -> run (ev);
      lastCalled = current;
 } while (exitflag);
}
```

Listing 2 DRIVER.HPP

```
#ifndef __DRIVER
#define __DRIVER

struct event {
 char type;
 int i1, i2, i3
};

#define KEYIN 0
#define HOTKEY 1
#define CALL 2

class proc {
 friend class handler;

 protected:
 int RETURNABLE, returnTo;
 handler* handlerp;

 public:
 int id;
 virtual void init (void);
 virtual void run (event&);
 virtual void done (void);

 proc (int = 1);
};

#define MAXPROC 200
#define EVENTSTACK 10

class handler {
 protected:
 int keyvec [0x200];
 proc *prvec [MAXPROC];
 int koliko;
 int current;
 virtual event getevent (void);
 virtual int event2proc (event&);
 event eventStack [EVENTSTACK];
 int eventStackP;

 public:
 int exitflag;
 handler (void);
 handler& operator + (proc&);
 handler& operator = (int);
 void insertEvent (event&);
 void run (int);
};

#endif
```

Listing 3 DEMO.CPP

```
#include <io.h>
#include <string.h>
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <fcntl.h>
#include "driver.hpp"

#define ALT1 0x178
#define ALT2 0x179
#define ALT3 0x17a
#define F5 319
#define ALTE 0x126
#define ESCKEY 27
#define DOWNKEY 336
#define LEFTKEY 331
#define RIGHTKEY 333
#define PGUPKEY 329
#define PGDNKEY 337

#define TABSIZE 4

int mode = 0, height;


// Klasa za pregled tekstualnih fajlova

class viewer: public proc {

 #define SIZE 10000
 char mem [SIZE];
 int items;
 #define LINES 1000
 char *line [LINES];

 char name [15];
 int x1, x2, y1, y2, x, y;
 void drbox (char *s);
 void fillbox (void);
 char *expand (char *to, char *from);
 void dosbox (int, int, int, int);

 public:
 viewer (char *fname, int, int, int, int);
 virtual void init (void);
 virtual void done (void);
 virtual void run (event&);
};


// naslednik klase proc: signalizira izlazak iz programa

class :public proc {
 virtual void run (event&) {handlerp -> exitflag = 0;}
} exiter;


// Klasa koja menja stanje ZOOM indikatora
```

Nastavlja se na sledećoj strani

Listing 3 DEMO CPP (nastavak)

```
class zoomclass: public proc {
 char buffer [4000];
 virtual void run (event&);
} zoomer;

// ****** Implementacija klase zoomclass ******

void zoomclass::run (event&)

{
 #pragma warn -pia
 if (zoom = !zoom)
 #pragma warn +pia
      gettext (1, 1, 80, 25, buffer);
 else
      puttext (1, 1, 80, 25, buffer);
 event ev = {CALL, returnto, 1d, 0};
 handler -> loadevent (ev);
}

// ****** Implementacija klase viewer ******

viewer::viewer (char *fname, int left, int top, int right, int bottom): proc ()

// Konstruktor klase viewer - ucitava datoteku
// postavlja pointere i crta prozor na ekranu

{
 int handle, bytes, i;
 char *p;

 strcpy (name, fname);
 if ((handle = open (fname, O_RDONLY|O_TEXT)) == -1) {
      perror ("Error: ");
      exit (1);
 }
 bytes = read (handle, (void*)mem, 8192);
 close (handle);
 *(mem + bytes++) = '\0';
 x = y = lines = 0;
 p = line [0] = mem;
 for (i = 0; i < bytes; i++, p++) {
      if (*p == '\n')
           line [++lines] = p + 1;
 }

 x1 = left; y1 = top; x2 = right; y2 = bottom;
 drbox ("\xda\xc4\xbf\xb3\xc0\xc4\xd9");

 fillbox ();
}

void viewer::init (void)

// Crta okvir s debelim linijama i popunjava ga

{
 drbox ("\xc9\xcd\xbb\xba\xc8\xcd\xbc");
 fillbox ();
}

void viewer::done (void)

// Crta okvir s tankim linijama

{
 drbox ("\xda\xc4\xbf\xb3\xc0\xc4\xd9");
}

void viewer::run (event& ev)

// *dlaval* dec prem s: ta osnovu pritisnute
// dirke menja sadrzaj ekrana

{
 if (ev.type == KEYIN) {
      switch (ev.i1) {
           case RIGHTKEY:x += 10; break;
           case LEFTKEY:if (x) x -= 10; break;
           case UPKEY:y --; break;
           case DOWNKEY:y++; break;
           case PGUPKEY:y -= height - 1; break;
           case PGDNKEY:y += height - 1; break;
      }
      if (y + height > lines)
           y = lines - height
      if (y < 0)
           y = 0;
      fillbox ();
 }
}
void viewer::drbox (char *s)

// Crta okvir na ekranu sa zadatim simbolima
// na pozicijl (x1 y1)-(x2 y2)

{
 int x1, y1, x2, y2;
 doscoo (x1, y1, x2, y2);
 height = y2 - y1 - 1;
 gotoxy (x1, y1);
 putch (*s++);
 for (int i = x1 + 1; i < x2; i++)
      putch (*s);
 putch (*++s);
 s++;
 for (i = y1 + 1; i < y2; i++) {
      gotoxy (x1, i);
      putch (*s);
      gotoxy (x2, i);
      putch (*s);
 }
 gotoxy (x1, y2);
 putch (*++s);
 s++;
 for (i = x1 + 1; i < x2; i++)
      putch (*s);
 putch (*++s);
 gotoxy (x1 + 2, y1);
 printf (" %s ", name);
}

char *viewer::expand (char *to, char *from)

// Prekopira; sa from na to, pri tome
// "prosirujuci" TABove. Vraca to.

{
 int i = 0;
 char *s = to;

 do {
      if (*from == '\t') {
           do {
                i++;
                *s++ = ' ';
           } while (i%TABSIZE);
      }
      else {
           i++;
           *s++ = *from;
      }
 } while (*from++ != '\0');

 return to;
}

void viewer::fillbox (void)

// Upisuje tekst gledanog fajla u okvir

{
 int x1, y1, x2, y2;
 doscoo (x1, y1, x2, y2);
 int width = x2 - x1 - 1;
 int yp = y, ys = y1 + 1;
 char buff [80], *s, *q, exp [500];

 do {
      memset (buff, ' ', width);
      *(buff + width) = 0;
      if (yp < lines) {
           s = expand (exp, line [yp]);
           for (int i = 0; i < x; i++) {
                if (*s == '\n')
                break;
                s++;
           }
           q = buff;
           for (i = 0; i < width; i++) {
                if (*s == '\n')
                break;
                *q++ = *s++;
           }
      }
      yp++;
      gotoxy (x1 + 1, ys);
      cputs (buff);
 } while (++ys < y2);
}

void viewer::doscoo (int& t1, int& t2, int& t3, int& t4)

// Odlucuje na osnovu prom. zoom da li koordinate
// x1 y1 x2 y2 predstavljaju ceo ekran ili samo prozor

{
 if (zoom) {
      t1 = 1; t2 = 1; t3 = 80; t4 = 24;
 }
 else {
      t1 = x1; t2 = y1; t3 = x2; t4 = y2;
 }
}

main ()

{
 _setcursortype (_NOCURSOR);
 viewer drhpp ("driver.hpp", 51, 1, 80, 24),
        drcpp ("driver.cpp", 1, 1, 50, 15),
        demo ("demo.cpp", 1, 16, 80, 24);
 handler disp;
 disp     + drhpp - ALT2
          + drcpp - ALT1
          + demo - ALT3
          + zoomer - F5
          + exiter - ALT-X;
 disp.run (drcpp, 1d);
 _setcursortype (_NORMALCURSOR);
 gotoxy (1, 25);
 return 0;
}
```

struct event {
  char type,
  int i1, i2, i3,
}

Polje **type** služi za opis događaja i može da sadrži tri predafinisane vrednosti KEYIN, HOTKEY i CALL. KEYIN označava da je korisnik pritisnuo bilo koju dirku. HOTKEY pokazuje da je pritisnuta neka dirka koja služi da pozove neku od procedura (kao što F10 poziva meni u TC++-u), a CALL služi za međusobne pozive procedura. Polja **i1**, **i2** i **i3** služe za prenos samog događaja. U slučaju pritisnute dirke **i1** sadrži njen ASCII kod, a u slučaju poziva **i1** i **i2** sadrže šifre pozvane i pozivajuće procedure, a **i3** sadrži neku proizvoljnu informaciju koja se prenosi. Daljim proširenjima programa mogla bi se uvesti i podrška za miša, pa bi se uvelo novi tip, recimo MOUSE, a **i1**, **i2** i **i3** bi sadržavale informacije o položaju miša i tipu klika (jedan klik, dupli ili držanje dugmeta na mišu). Kao što se vidi, ova tri **int**-a su sasvim dovoljna da prenesu informacije potrebne za prenos događaja, ali u slučaju da za neku buduću aplikaciju zafali mesta, u njih se uvek može upisati pointer na neku dužu strukturu koja detaljnije opisuje događaj.

Klasa **proc** sastoji se od konstruktora same klase i tri virtualne funkcije. Iako se one mogu pozivati i direktno, njihova svrha jeste u tome da ih poziva klasa **handler** kada utvrdi da je to potrebno. Funkciju init **handler** poziva svaki put kada se izvršavanja neke druge procedure prelazi na ovu. Funkciju **run** **handler** poziva svaki put kada se desi neki događaj koji se odnosi na datu proceduru, a **done** biva pozvana svaki put kada se sa date procedure prelazi na sledeću. Tok programa dakle izgleda ovako

```
proc1 init
proc1 run       // korisnik pritisnuo dirku
proc1 run

proc1 done      // prelazak na novu proc
proc2 init
proc2 run
proc2 run
proc2 done
proc3 init
```

Važno je primetiti da ni jedna od ovih funkcija ne radi ništa – klasa **proc** je tzv. apstraktna bazna klasa koja služi kao kostur za buduće naslednike. Na gornjem primeru klase **proc1**, **proc2** i **proc3** ne predstavljaju instance klase **proc**, već instance klasa koje nasleđuju **proc**.

Od podataka klasa **proc** sadrži sledeće promenljive:

- **id** – svaka procedura ima svoj „redni broj" pomoću koga se poziva
- **handlerp** – pointer na instancu klase **handler** na koju je klasa prijavljena
- **RETURNABLE** – signališe da li pozvana procedura može da vrati kontrolu proceduri koja ju je pozvala. Na primeru integrisane okoline TC++-a, editorski prozori imaju atribut RETURNABLE postavljen, a dijalozi nemaju. Ako iz editorskog prozora 1 sa F3 pozovemo **File/Load** dijalog, a on pozove editorski prozor 2, zatvaranjem prozora 2 kontrola će se vratiti editorskom prozoru 1, a ne dijalogu.
- **returnTo** – sadrži **id** procedure na koju se treba vratiti. (Na gornjem primeru **returnTo** editorskog prozora 2 sadrži **id** editorskog prozora 1.)

Kako je klasa **handler** nešto komplikovanija, neću do detalja opisivati njenu strukturu. Ukratko, sastoji se od:

- steka **eventStack**, koji sadrži događaje koji „čekaju na red" da budu obrađani. U principu služi za pozive među procedurama. Događaji se stavljaju na stak funkcijom **insertEvant**.
- funkcije **getevent**, koja u slučaju da nema ničega na steku vraća pritisnutu dirku sa tastature
- funkcije **event2proc**, koja na osnovu događaja dobavljenog sa **getevent** odlučuje na osnovu interne tabele koju proceduru treba pozvati
- operatora „+", koji instanci klase **handler** prijavljuje koje se sve procedure nalaze u programu
- operatora „," koji postavlja vektor interne tabele dirki na zadnju prijavljenu proceduru
- funkcije **run**, koja kao parametar prima **id** procedure u koju se prvo ulazi. Ova funkcija zapravo poziva po potrebi funkcije klase **proc**
- **exitflag** je zastavica koja pokazuje da li program treba i dalje da radi. Kada neka od procedura odluči da prekine program, dovoljno je da sa

  handlerp -> exitflag = 0;

  postavi **exitflag** na 0 i po njenom završetku program će stati.

Ovakva kakva jeste, klasa **handler** je u potpunosti funkcionalna. Međutim, za razvoj nekog komercijalnog programa morala bi se proširiti barem podrškom za miša. Tu do izražaja dolazi C++: bilo kakva dalja proširenja mogu se vrlo lako ostvariti zamnom virtualnih funkcija **getevent** i **event2proc**. U slučaju miša, dodao bi se novi tip događaja MOUSE, napravila bi se matrica koja bi predstavljala ekran, a sadržala bi vektore na procedure. Uz nekoliko desetina linija koda dodala bi se cela podrška za miša. Na isti način bi se mogla dodati podrška za COM portove, interapte ili bilo šta drugo.

JEDNOSTAVNI DEMO

Uz biblioteku DRIVER priložen je i mali demo program koji služi za pregled većeg broja tekstualnih datoteka u isto vreme. Sastoji se od tri klase:

- Klasa **viewer** služi za sam pregled datoteka. U njoj funkcija **init** ima zadatak da prozor sa tekstom datoteke stavi na „vrh" ekrana i da oko njega iscrta okvir sa duplim linijama. Funkcija **done** crta isti taj okvir, ali sa tankim linijama, signališući da je prozor neaktivan, dok funkcija **run** na osnovu kursorskih i **PgUp/PgDn** tastera vrši pomeranje virtualnog prozora po tekstu. Tri instance klase viewer služe za pregled datoteka DRIVER.CPP, DRIVER.HPP i samog demo programa DEMO.CPP.
- Bezimena klasa sa instacom **exiter** služi za izlazak iz programa.
- Klasa **zoomclass** omogućuje zum tj. maksimizaciju datog prozora preko celog ekrana.

Kada sam pisao demo, nisam imao klasu **zoom** u planu. Međutim, shvativši koliko ju je lako implementirati, nisam mogao da odolim a da ja ne uključim. Dirke su pridružene svakoj od insanci ovih klasa tako da program što više liči na integrisanu okolinu TC++-a. Između prozora možete se „šetati" pomoću dirki ALT-1, ALT-2 i ALT-3, zum se ostvaruje pritiskom na F5, a izlazi se iz programa pritiskom na ALT-X. Iako je ovaj demo nema drugu svrhu osim pregleda datoteka koje ga čine, ipak se pozabavite njima: uz minimalne izmene (desetak radova), on može postati opšti program za pregled velikog broja datoteka.

NOVI HORIZONTI

Iako je ED programiranje nastalo iz potrebe za smanjenjem složenosti programa, sada kada je ova tehnika široko prihvaćena, otvaraju se novi horizonti u programiranju. Multitasking, koji je godinama predstavljao veliku muku pod DOS-om, kod ED programa je trivijalan za izvođenje. Kako je sva komunikacija sa korisnikom izdvojena u jednu funciju (**handler::getevent**), dovoljno je u nju ugraditi mahanizam koji kada se ništa ne dešava poziva razne funkcije (ili procedure) koje rade u pozadini. Na taj način se ostvaruje tzv. kooperativni multitasking, koji ne zahteva petljanja sa stekom, interaptima i drugim mašinski zavisnim elementima, tako da može da radi bez obzira na ciljnu mašinu. Ko je video komunikacioni program *TeleMate*, prepoznaće ovde upravo taj interni multitasking, koji ovaj program čine izborom broj jedan velike većine korisnika modema.

Sve u svemu, iako ED programiranje nije rešenje za sve programerske dileme, ono predstavlja jedan evolucioni korak napred ka boljem pisanju programa kao što je to učinilo strukturno ili objektno programiranje. Visoko preporučeno.

PREUZMITE SA SEZAMA

Izvorni i izvršni kod programa opisanih u ovom tekstu može se preuzeti sa *Sezama* iz direktorijuma r:\rsoft\ pod imenom r077drv.zip

ŠTA IMA NOVO



Ima li joj leka: Inteligentna hardverska zaštita SmartKey

Zaštite

SMARTKEY-II/EP

SmartKey II EP je naziv novog modula za hardversku zaštitu autorizovanog softvera koji raspolaže ugrađenim sistemom za praćenje i notiranje svakog pokušaja ilegalnog pristupa. Modul raspolaže sa 16 bajtova namenjenih za programabilnu lozinku i 64 bajta sigurnosne read/write memorije koja služi za smeštanje ličnih informacija, kao što su serijski broj, ima klijenta, pretplatni period i pristupna lozinka. Po završenom programiranju, sve informacije se zamrzavaju kako bi se sprečila neželjena manipulacija unetim podacima.

SmartKey ograničava upotrebu programa na rad sa samo jednom kopijom ali ne sprečava pravljenje back-up kopija. Instalira se na paralelni port između računara i štampača, a u svaki modul se upisuje jedinstvena šifra. Pored toga, omogućeno je i enkriptovanje posebno poverljivih informacija. Cena: 85.000 lit.

Kontakt adresa: Eutron S.r.L., Security Division, Via Gandhi 12, 24048 Treviolo BG, Italy, tel. 39-35-690.018, fax 39-35-692-229

ZABLUDE O 486

Nastavak sa strane 12

Čim instalirate QEMM, 386MAX, EMM386, neki novi *cache* program ili nešto slično (a za šta bi služio 80386/80486 PC na kome nema nekog od ovih programa?), vaš mikroprocesor uopšte nije u realnom modu. Samo se uspešno pretvara – onih 640K memorije koje vidite možda uopšte nisu na početku adresne mape, a možda čak i nisu fizički kontinualni; *high* memorija je skrpljena od parčića, svaki pristup RAM-u ide preko specijalnih drajvera da bi se simulirala LIM memorija... Pokušajte, na primer, da upišete nešto u „interesnu zonu QEMM-a" – poruka od par redova i vaš program je suspendovan! Ili još jednostavnije – setite se onoga što ste naučili na starim mikroprocesorima pa napunite status registar željenim sadržajem koristeći sekvencu MOV EAX, *vrednost*; PUSH EAX; POPFD i sve će lepo proći. Pa ipak, pogled u statusni registar će pokazati da je upisano samo ono što je operativni sistem smatrao da smete upisati – ako je neki fleg van vašeg domašaja, njegovo stanje se uopšte nije promenilo!

Ukratko, stavljen vam je na raspolaganje samo jedan virtualni PC koji je prividno pod vašom kontrolom, a u stvari postoji nevidljiva „viša sila" koja upravlja stvarima. Pazite, vaš 80486 vas posmatra!

KATALOŠKO STABLO

Više puta smo isticali jednu od glavnih osobina *Windows* okruženja – prijateljski odnos prema korisniku. Dobra strana *Windows* korisničkog interfejsa je što standardizuje mnoge rutinske postupke u radu sa programima. Korisnika ovo oslobađa potrebe da za svaki novi program uči pravila rukovanja iz početka, a programeru smanjuje posao oko kreiranja „važnih sporednih" stvari u programu, i ostavlja mu više vremena da se bavi osnovnom funkcijom programa. Nije onda čudo što ćemo još pažnje posvetiti tehnikama programiranja koje se odnose na korisnički interfejs *Windows* aplikacija.

Dejan Dinčić

Novi primer poslužiće da razmotrimo funkciju koja se često javlja u različitim programima: promena vidljivog dela slike ili teksta – popularnije skrolovanje. Problem se javlja uvek kada celovit prikaz premašuje raspoloživu površinu. Tada je potrebno prikazati samo jedan deo slike, a na zahtev korisnika „pomeriti" sadržaj i učiniti vidljivim ono što se pružalo preko ivica ekrana ili prozora.

U *Windows* programu skrol ima dva aspekta. Prvi je zahtev za skrolovanjem, koji dolazi sa poznatog **scrollbar**-a, a drugi je realizacija tog zahteva – pomeranje jednog dela prikaza i iscrtavanje drugog dela.

KLASA Scrollbar

Već smo pisali o predefinisanim klasama prozora. Tada je bilo reči i o tome da se predefinisane klase (**listbox button edit**) mogu naći u dijalogu i tada predstavljaju „kontrole" tog dijaloga (controls). Prozori predefinisane klase takođe mogu biti kreirani kao potprozori glavnog prozora (kao u našem ranijem primeru FileView). **Scrollbar** je jedna od predefinisanih klasa prozora i njena upotreba nije ograničena samo na grafičko skrolovanje. Ova vrsta prozora koristi se često za postavljanje raznih vrednosti koje se mogu predstaviti nekim linearnim opsegom vrednosti. Očigledan primer je program *Control Panel* (koji se dobija uz *Windows*) i njegova opcija **Mouse**. **Scrollbar** se tu koristi za određivanje brzine praćenja miša i brzine dvostrukog klika.

Mada ćemo se ovaj put zadržati na prvobitnoj primeni **scrollbar**-a, iz programske ilustracije biće sasvim jasno kako upotrebiti ovaj elemenat korisničkog interfejsa i u druge (možda egzotičnije) svrhe.

Zadatak koji smo ovaj put postavili pred nas *Windows* program nije mnogo nov ni originalan. Potrebna nam je bila atraktivnija i preglednija varijanta DOS naredbe **tree**. Grafički prikaz stabla direktorijuma obično prevazilazi površinu ekrana i tu se javlja potreba za skrolovanjem prikaza. U rešenju grafičkog prikaza do izražaja su došle dve stvari: upotreba MM_ANISOTROPIC načina za mapiranje ispisa u grafički kontekst prozora i upotreba **scrollbar**-a za korisnikovu kontrolu vidljivog dela stabla.

Pored uobičajenog korišćenja **scrollbar**-a kao potprozora na proizvoljnom mestu u programu ili u dijalogu, autori *Windows*-a su valjda zbog česte upotrebe predvideli još jedan način za njihovo korišćenje. Zapravo najjednostavniji način se sastoji u tome da se kod kreiranja glavnog prozora mogu navesti atributi WS_HSCROLL i WS_VSCROLL, što će prouzrokovati automatsko pojavljivanje horizontalnog/vertikalnog **scrollbar**-a duž ivica prozora. Tako kreiran **scrollbar** je sistemski vezan za prozor roditelj. Poruke o korisnikovoj intervenciji u tom području prenose se prozoru roditelju.

Poruka WM_HSCROLL šalje se prozoru kada korisnik „klikne" na horizontalni **scrollbar**, a poruka WM_VSCROLL kod „klika" na vertikalni. Prvi parametar (**wParam** u *Windows* notaciji) nosi **scrollbar** kod koji bliže određuje poruku. Kod može imati sledeće vrednosti:

Listing 1





Kreiranje prozora i obrada poruka

Listing 2

```
/* ... ekrana pomocu ... [illegible] */
void WTreePaint (HWND hwnd, HDC hDC)
HWND  hWnd;
HDC   hDC;
{
    RECT  Rect;
    [illegible]
    HANDLE  hNewFont,
            hOldFont;

    hDC = hdc;

    SetMapMode(hDC, MM_ANISOTROPIC);
    SetWindowExt(hDC, [illegible]);
    SetViewportExt(hDC, [illegible]);

    GetClientRect(hWnd, &Rect);
    SetViewportOrg(hDC, [illegible]);
    SetViewportExt(hDC, Rect.right, Rect.bottom);

    hNewFont = GetStockObject(ANSI_FIXED_FONT);
    hOldFont = SelectObject(hDC, hNewFont);

    if (root != NULL)
        DrawTree(root, ROOT);

    SelectObject(hDC, hOldFont);
    DeleteObject(hNewFont);
}

/*-----------------------------------------------*/
/*  [illegible]                                  */
/*-----------------------------------------------*/
void DrawDir (name, lvl, ypos)
char  *name;
int   lvl;
int   ypos;
{
    short x, y;

    x = ypos*(X_WIDTH+[illegible]);
    y = [illegible];
    Rectangle(hDC, x, y, x+X_WIDTH-1, y+Y_HEIGHT-1);

    TextOut(hDC, x+2, y+2,
            (LPSTR) name,
            (short) strlen(name));
}

/*-----------------------------------------------*/
/*  crta [illegible] stabla                      */
/*-----------------------------------------------*/
void DrawTree (prev
PDIR  prev;
{
    PDIR  wp, last;

    for (wp = prev->down; wp != NULL; wp = wp->next)
    {
        if (wp [illegible])
            if (wp == prev->down)
            {                                   /* prvi [illegible] */
                MoveTo(hDC, prev->x, [illegible]);
                LineTo(hDC, wp->x, [illegible]);
            }
            else
            {                                   /* [illegible] */
                MoveTo(hDC, [illegible]);
                LineTo(hDC, [illegible]);
            }
        else
            if (wp != prev->down)
            {                                   /* [illegible] */
                MoveTo(hDC, [illegible]);
                LineTo(hDC, [illegible]);
            }
            else
            {                                   /* [illegible] */
                MoveTo(hDC, [illegible]);
                LineTo(hDC, [illegible]);
            }

        if ([illegible])
            DrawDir(wp->name, wp->y, wp->x);
        else
            DrawDir(wp->name, wp->x, wp->y);
        DrawTree(wp);
        last = wp;
    }
}
```

```
/*-----------------------------------------------*/
/*  formira stablo                               */
/*-----------------------------------------------*/
int FormTree (path, lvl, up, xp)
char   *path;
int    lvl;
HANDLE up;
int    xp;
{
    int    bd, vb;
    int    code;
    int    first;
    char   mpath[[illegible]];
    struct find_t  [illegible];
    HANDLE prev;

    strcpy (mpath, path);
    strcat (mpath, "[illegible]");

    first = 1;
    bd = [illegible];
    [illegible]
    code = _dos_findfirst(mpath, _A_SUBDIR, &statr);
    while (!code)
    {
        if ((statr.attrib & _A_SUBDIR) [illegible])
        {
            /* dodaj dir u stablo */
            if (first)
            {
                /* [illegible] */
                prev = AddDir (statr.name, [illegible]);
            }
            else
                prev = AddDir (statr.name, [illegible]);
            [illegible]
            [illegible]
            strcpy(mpath, path);
            strcat(mpath, statr.name);
            strcat(mpath, "\\");
            if (lvl [illegible] MAX_LVL)
                vb = FormTree(mpath, lvl+1, prev, [illegible]);
            bd++;
            if (vb > [illegible])
                bd = vb-1;
            first = 0;
        }
        code = _dos_findnext(&statr);
    }
    [illegible]
    return(bd);
}

/*-----------------------------------------------*/
/*  [illegible] zahtev za scroll                 */
/*-----------------------------------------------*/
void scroll (code, h, dir)
WORD  code;
int   *h;
int   dir;
{
    int  step;
    int  maxx;
    RECT Rect;

    if (dir == [illegible])
    {
        step = [illegible];
        maxx = MAX_X-VIS_X;
    }
    else
    {
        step = [illegible];
        maxx = MAX_Y-VIS_Y;
    }

    switch (code) {
      case SB_TOP:
        *h = 0;
        break;
      case SB_LINEDOWN:
      case SB_PAGEDOWN:
        *h += step;
        break;
      case SB_LINEUP:
      case SB_PAGEUP:
        *h -= step;
        break;
    }
    *h = abs([illegible]);
    [illegible]
}
```

Formiranje i iscrtavanje stabla



Kodovi uz poruke **WM VSCROLL** i **WM HSCROLL koji detaljnije opisuju zahtev korisnika**

SB BOTTOM	Skrol do dna (dole ili desno)
SB_ENDSCROLL	Kraj skrola (ako je dugme miša bilo duže bilo pritisnuto, zatim oslobođeno)
SB LINEDOWN	Skrol jedna linija dole (desno)
SB_LINEUP	Skrol jedna linija gore (levo)
SB_PAGEDOWN	Skrol jedna strana dole (desno)
SB_PAGEUP	Skrol jedna strana gore (levo)
SB_THUMBPOSITION	Skrol na absolutnu poziciju
SB_THUMBTRACK	Kontinualni skrol („vučenjem" miša) do neke pozicije
SB_TOP	Skrol na vrh (gore ili levo)

Termini „jedna linija" i „jedna strana" su samo logičke kategorije uvedeni da se može razlikovati zahtev za skrolovanjem u manjim ili većim koracima. Koliko ta „linija" ili „strana" iznosi zavisi naravno od prirode aplikacije ili prikaza. Ako se radi o prikazu teksta, onda program i može stvarno da interpretira zahtev pomeranjem za liniju ili stranu teksta. U slučaju nekakve grafike „linija" može biti jedna ekranska tačka (piksel) ili već neka druga logička jedinica. Slika 1 ilustruje kako kod koji se dobija uz poruku zavisi od mesta na **scrollbar** u gde je korisnik „kliknuo" mišem.

Činjenica da aplikacija odlučuje o značenju pojma „linija" i „strana" vodi nas do sledeće bitne osobine podele posla. *Windows* će istina nacrtati **scrollbar** ove u našem prozoru, javiće mu porukama (WM_HSCROLL i WM VSCROLL) kada korisnik želi nešto da pomeri, ali *Windows* neće baš ništa uraditi da pomeri sliku ili tekst koji je prikazan u prozoru. Naš program mora sam da odluči kako će prikazati novi, osvežen i ispravan sadržaj s obzirom na zahtev za skrolovanjem.



Rezultat rada programa *Wtree*

Najprostiji način da se ovo uradi jeste da se izračuna šta treba sad da bude vidljivo i da se cela površina prozora ponovo iscrta (vidljivi deo prikaza). Malo naprednija tehnika podrazumeva i korišćenje sistemskih funkcija **ScrollWidow** ili **ScrollDC**. Ove funkcije imaju smisla samo ako deo starog sadržaja treba da ostane vidljiv na ekranu i posle skrol operacije (to jest, ako je skrol manji od pune strane). Obe funkcije izvršavaju fizičko „skrolovanje" površine prozora.

Nastavija se na strani 61

Zoran Životić

PRINTF

Do sada sam već nekoliko puta pominjao funkcije **printf/scanf**. Ovim funkcijama ću se sigurno vraćati i u budućem jer dosadašnjim objašnjenjima nisam pokazao sve njihove mogućnosti. Evo, dakle, još jedne od korisnih stvari.

Sasvim jednostavan oblik formatskog stringa kojim zadajemo da neka numerička promenljiva bude ispisana u formalizovanom obliku, recimo

double f;
printf("%10.2lf\n",f);

postaje nepogodan u situacijama kada ne znamo unapred dimenzije polja u koje želimo da „uguramo" ispis. Parametar „10 znakova sa preciznošću od 2 decimale" je u ovoj kombinaciji kodiran u sam program i ne može se menjati prema potrebama. Na prvi pogled, postoji jednostavno rešenje problema – formatski string možemo dinamički formirati na sledeći način:

```
void fmtout(double f, int w, int dp)
{
    char fmts[32];
    sprintf(fmts,"%%%d.%dlf\n",w,dp);
    printf(fmts,f);
}
```

U stringu **fmts** smo formirali formatski string prema vrednostima širine polja (**w**) i broja decimalnih mesta (**dp**) (dvostruki znak %% označava da se radi o samom znaku %, a ne prefiksu formata) i zatim taj string upotrebili kao formatski string za ispis promenljive **f**.

Rešenju se ništa ne može prigovoriti sem činjenice da postoji jednostavniji način, koji je sadržan u samoj **printf** funkciji. Unutar formatske sekvence znak * ima posebno značenje:

printf("%*.*lf",w,dp,f);

Umesto konkretne vrednosti parametra možemo staviti znak * i time naterati funkciju da njegovu vrednost potraži među parametrima same **printf** funkcije. Naravno, parametar mora biti u tom trenutku interpretacije formatskog stringa „prvi sledeći na listi" i treba da bude **int** tipa.

Printf funcija krije u sebi još mnogo iznenađenja. Do koje mere se radi o razređenoj funkciji pokazaće i jedan stvarno egzotičan oblik formatske komande. Kao što je poznato, funkcija vraća **int** vrednost koja nam ukazuje koliko je znakova stvarno ispisano (dužina formiranog stringa) ili na grešku u ispisu ukoliko je vrednost manja od nule. Za premeravanje dužine delimičnog ispisa možemo upotrebiti

printf("%s%n%s",s1,&len1,s2);

Nakon izvršavanja na ekranu ćemo imati ispis „[sadržaj stringa s1][sadržaj stringa s2]", a u **int** promenljivoj **len1** će se naći broj znakova koji je ispisan do mesta na kome se **%n** nalazi u formatskom stringu. To konkretno znači da smo u ovom slučaju dobili dužinu stringa **s1**. Oblik **%n** se zaista retko koristi (verovatno i zbog toga što nije dovoljno poznat) jer se obično pre samog ispisa nizom funkcija proveri šta će se sa ispisom desiti. Ipak, imajte na umu i ovaj oblik – doći ćete sigurno u situaciju kada ćete moći korisno da ga upotrebite.

SCANF I LIB

U jednom od prvih tekstova iz ove serije bilo je reči o problemu poruke tipa „Run time error: Floating point support not loaded." Ukratko, ukoliko kompajler nigde ne detektuje upotrebu floating-point promenljivih, deo koda iz biblioteke neće biti linkom ubačen u program. Jedno od mesta na kome se ne može detektovati upotreba je **scanf** funkcija – zbog promenljivog broja parametara kompajler se ne upušta u njihovu analizu (niti može bez analize samog sadržaja formatskog stringa znati tipove parametara), pa otuda ova floating point operacija (dodela vrednosti putem formatizovanog čitanja) ostaje neprimećena i bez potrebnog koda iz biblioteke. Naravno, teško je zamisliti situaciju u kojoj imamo, recimo, samo

scanf("%f",&f);

a da se pri tom promenljiva f više nigde ne upotrebljava u nekom „normalnom" kontekstu (ukoliko se ne radi o nekom eksperimentu sa ispitivanjem binarnog sadržaja float promenljivih i slično). Zato problemu i ne bih posvetio pažnju da on ne nameće jedno drugo interesantno pitanje.

Pitanje glasi: „Kako linker zna kada treba deo iz biblioteke povezati u osnovni program?" Odgovor je naravno jednostavan – kada nađe na nepoznat simbol (bilo promenljive, bilo funkcije) linker pretražuje biblioteku i ukoliko ga tamo pronađe, ceo modul u kome se taj simbol nalazi povezuje sa programom. Sad dolazi ključno pitanje – kako kompajler uopšte može naterati linker da poveže floating point (FP) podršku u slučaju **scanf** funkcije, kada nigde ne figuriše nikakav simbol koji bi trebalo pronaći u biblioteci? Da kojim slučajem u program ubacimo

f = p/12.1;

kompajler bi odmah detektovao da će FP deo biti potreban i **scanf** će korektno nakon toga raditi, ali i dalje se ne vidi kako kompajler javlja linkeru šta je potrebno, kad ni ovde nemamo nikakav simbol koji bi trebalo pronaći u biblioteci. Da bi se odgovorilo na ovo pitanje, potrebno je poznavati način na koji je biblioteka organizovana.

Da bi neki modul postao deo biblioteke, pred njega se ne postavljaju nikakvi posebni zahtevi – dovoljno je da na uobičajen način napišemo bilo kakvu funciju, da je prevedemo u .OBJ datoteku i da je zatim LIB programom ubacimo u postojeću biblioteku ili sa tim modulom kreiramo novu. Ako se naš program zvao, recimo, FILEUTIL.C i ako je u njemu neka funkcija **addtex()**, onda će ovaj modul (FILEUTIL.OBJ) biti ubačen u biblioteku, a linker će ga povezati sa programom ukoliko u njemu pronađe poziv **addtex()** funkcije.

Ukoliko FILEUTIL.C sadrži više funkcija, na pojavu bilo koje od nje **čitav** FILEUTIL.OBJ će biti povezan sa programom. Potpuno ista stvar se dešava sa podacima. Ako je našoj funkciji **addtex()** potrebna neka globalna promenljiva (recimo **char filename[80]**), onda će svako pozivanje na ovu promenljivu u bilo kom programu naterati linker da opet poveže **ceo** FILEUTIL.OBJ kod.

Biblioteke se zato najčešće organizuju tako da srodne funkcije budu grupisane u jedan .OBJ modul jer se time štedi vreme pri linkovanju. Sasvim je logično da će nakon **fopen** program sigurno izvesti i **fclose**, pa se obe funkcije mogu staviti u jedan modul i tako uštedeti deo vremena linkeru na pretraživanju biblioteke.

Da budem malo konkretniji i da dam postupak za kreiranje i ubacivanje u biblioteku jedne korisne funkcije. Naziv **addext()** može da posluži kao naziv funkcije kojoj će biti prosleđen

Listing 1 Funkcija addext

```
#include <stdio.h>
#include <string.h>

char *addext(char *fname, char *fext)
{
    char *dot;

    if ( (dot=strchr(fname,'.')) != NULL )
        strcpy( dot+1, fext);
    else
    {
        strcat( fname, ".");
        strcat( fname, fext);
    }
    return fname;
}

/* primer upotrebe

    f = fopen( addext(argv[0],"cfg"), "r" );
*/
```

Listing 2 Funkcija copyfile

```
#include <stdio.h>
#include <malloc.h>

int copyfile(char *source, char *destination)
{
    char *buf;
    FILE *fs,*fd;
    size_t rb;

    if ((fs=fopen(source,"rb"))==NULL)
        return 1;   /* source file not found */
    if ((fd=fopen(destination,"wb"))==NULL)
    {
        fclose(fs);
        return 2;  /* unable to create destination file */
    }
    if ((buf=malloc(0x8000))==NULL)
    {
        fclose(fs);
        fclose(fd);
        return 3; /* not enough memory for buffer */
    }
    while ( (rb=fread(buf,1,0x8000,fs)) != 0)
        if ( fwrite(buf,1,rb,fd) != rb )
        {
            fclose(fs);
            fclose(fd);
            free(buf);
            unlink(destination);
            return 4;    /* disk full */
        }
    fclose(fs);
    fclose(fd);
    free(buf);
    return 0;    /* ok */
}
```



naziv datoteke i željena ekstenzija, a on će formirati pun naziv, uklanjajući, ako je to potrebno, već postojeću ekstenziju (listing 1).
Postupak za ubacivanje u standardnu biblioteku (MSC) za SMALL model je sledeći:

CL -AS -Gs -Ox -c fileutil.c
LIB \c600\lib\slibce +fileutil;

Nakon ovoga, **addext()** postaje ravnopravna bilo kojoj drugoj bibliotečkoj funkciji. Naravno, pri pisanju bibliotečkih funkcija treba imati na umu neka pravila „pristojnog ponašanja", ali globalno važi da se bilo koja obj datoteka može ubaciti u biblioteku, jer je biblioteka nije ništa drugo do organizovana kolekcija objektnih modula.

Pretpostavimo sada da nam je, nekim zaista neverovatnim povodom, potrebno da linker poveže ovu funkciju u naš program, a da se ona pri tome nigde ne poziva. Najjednostavniji način da to učinimo je dodamo jednu globalnu promenljivu, dakle da modul FILEUTIL.C započeno ovako:

```
#include <stdio.h>
#include <string.h>
int _linkfileutil;
char *addext(char *fname, char *fext)
{
```

Ako u bilo koji program ubacimo deklaraciju

extern int _linkfileutil;

i potom bilo gde u kodu neko referisanje na ovu promenljivu, recimo

_linkfileutil = 1;

funkcija **addtex()** će se naći u konačnom programu iako se nigde ne koristi – ona se jednostavno nalazi u istom modulu sa promenljivom i kao celina ulazi u .exe. Ako zađemo još dublje u sistem po kome linker radi, ustanovićemo da nije ni potrebno da stvarno diramo vrednost **_linkfileutil** promenjive – dovoljno je da u .obj datoteci postoji informacija da je ona potrebna, za šta bi teorijski bila dovoljna **extern** deklaracija koju smo naveli. C kompajler će pak ovu deklaraciju, kao jedan od metoda optimizacije, ignorisati ukoliko se navedena promenjiva stvarno ne koristi nigde u programu.

Ovo, naravno, ne važi za sam kompajler – **extern** deklaraciju može preneti u .obj datoteku po volji, pa to, da se sada vratimo na onaj **scanf** problem, zaista i čini. MSC za ovu namenu koristi promenjivu **int** tipa **_fltused**. Kada primeti da se u programu pojavljuju FP operacije, u .obj datoteku ubacuje nešto poput **extern _fltused**, što je linkeru dovoljno da poveže ceo modul u kome se ova promenjiva nalazi. Dakle, ako zaista imate potrebu da ovako prevarite MSC, evo načina:

```
extern int _fltused;
float f;
main()
{
    if (_fltused) ;
    scanf(„%f",&f);
}
```

Ne treba ni pominjati da je ovo rešenje potpuno neprenosivo, niti da ga uopšte treba uzimati ozbiljno. Nadam se ipak da je poslužilo kao dobra ilustracija link mehanizma.

FREAD/FWRITE

Funkcije za čitanje/zapisivanje niza blokova iz datoteka izazvale su nedavno malu nedoumicu na Sezamu. Funkcija **fread** ima sledeću deklaraciju:

```
unsigned int fread (void *buffer
                    unsigned int blok_size
                    unsigned int no_of_blocks
                    FILE *f);
```

Rezultat funkcije je broj uspešno pročitanih blokova. Šta se dešava ako imamo neku datoteku dugačku 123 bajta i pokušamo čitanje sa

fread(buffer,100,2,f);

Funkcija će vratiti broj 1, jer je toliko celih blokova od po 100 bajtova uspela da pročita. Pri tome nama način da saznamo da li nam je ostao nepročitan 1 ili 99 bajtova. Ovakva logika je sasvim ispravna kada se funkcija koristi tačno u onom smislu u kom je i definisana. Pretpostavka je da će **buffer**, u stvari, biti pointer na neki niz struktura. Recimo, neka je to neki slog **struct radnik** i tada će

lastradnik =
fread(ranici, sizeof(struct radnik), 100, f);

promenjiva **lastradnik** dobiti broj stvarno pročitan celih slogova. Da li je pri tome u datoteci ostao i neki višak bajtova potpuno je svejedno, jer bi takav višak samo delimično popunio naš slog i time bio neupotrebljiv.

Sasvim je drugačija situacija kada, recimo, kopiramo sadržaj neke datoteke – tada je jasno, svaki bajt važan i potrebno je da znamo tačno koliko je blokova pročitano i koliko parče bloka je ostalo. Zato treba i pisati upravo onako kakav je i smisao kopiranja, a to je

bytesread = fread(buffer,1,200,f);

Dakle, želimo da pročitamo 200 blokova od po jedan bajt – funkcija će nam vratiti koliko je stvarno pročitala blokova, odnosno bajtova. Glavno pitanja u ovom rezonovanju je ima li ikakve razlike u sledeća dva načina (sem, naravno, vrednosti rezultata funkcije):

fread(b,1,200,f);
fread(b,200,1,f);

Odgovor na pitanje je – nema. Naime, kompajler ne prevodi ovakav kod u neku internu petlju, u kom slučaju bi razlika u brzini izvršavanje bila drastična jer zaista nije svejedno da li se operativni sistem poziva jednom sa zahtevom za čitanje 200 bajtova ili 200 puta sa zahtevom za jednim bajtom. Bibliotečka funkcija množi ove dve vrednosti i zatim na osnovu ukupne potrebne dužina postavlja zahtev operativnom sistemu. Ovde se može desiti da zahtev preraste limit od 64K (u LARGE modelima), ali se C i tada odlično snalazi – ukupan zahtev se deli na manje blokove (MSC to radi na po 32K), ali ova podela na blokove i dalje nema veze sa vrednostima koje ste vi zadali. Kada operativni sistem vrati broj pročitanih bajtova, C funcija taj broj deli sa vašom dužinom bloka i tako stiže do vrednosti koju vraća.

Prema tome, bez ikakve opasnosti možemo pisati ono što po logici stvari odgovara situaciji. Našoj biblioteci i prethodnom primeru funkcije **addext()** možemo dodati i funkciju **copyfile** (listing 2).

KATALOŠKO STABLO

Nastavak sa strane 59

Sumirajmo: **scrollbar** samo dostavlja zahtev za skrolovanjem aplikaciji. Aplikacija se brine o tome da pomeri prikaz (teksta, grafike ...) ponovnim iscrtavanjem cele površine prozora, ili dela površine uz korišćenje *Windows* funkcija za stvarno pomeranje slike.

I još jedan detalj oko podela posla: indikator skrol pozicije se ne pomera sam, aplikacija mora da mu promeni poziciju nakon obrade zahteva za skrolovanjem. Iako na prvi pogled deluje kao višak posla i stvar koja je logično mogla da se automatizuje u okviru **scrollbar** definicije, ipak je primerenije aplikaciji da odredi da li će i kako promeniti poziciju indikatora. Ovome je namenjena sistemska funkcija **SetScrollPos**.



Drugi deo rešenja zadatka nalazi se u crtanju dela slike koji u trenutku prikaza važi kao vidljiv. Dobar deo rešenja zasniva se na upotrebi MM_ANISOTROPIC načina mapiranja. Ovaj „mapping mode" je vrlo fleksibilan: njime se može zadati odnos između koordinata u logičkom prozoru i koordinata u fizičkom prozoru. Više od toga, može se zadati proizvoljna orijentacija osa, i odrediti položaj koordinatnog početka u okviru prozora. Pomeranje koordinatnog početka po prozoru zvuči baš kao ono što nam treba! Ako ovo nakom zvuči malo konfuzno, zamislite sledeću situaciju: crtež na ogromnoj tabli (logički crtaći prostor). Kroz mali vizir (fizički prozor na ekranu – *viewport*) vidi se samo mali deo slike. Kad gledalac poželi da vidi ono što je malo više desno jednostavno se pomeri desno i vidi drugo parče slike. Ovo pomeranje desno se u programu izvodi funkcijom **SetWindowOrg**, uz predaju nove vrednosti X za koordinatni početak. Ova analogija je sasvim dovoljna za onaj deo mogućnosti koji smo koristili u *Wtree* programu. Napomenimo da je još moguće promeniti i veličinu prostora koja se vidi kroz vizir, što stvara utisak zumiranja. Sve ovo se u našem programu dešava kao reakcija na rukovanje **scrollbar**-ovima.

FUNKCIJA FormTree

Na kraju, da pojasnimo i način na koji smo došli do stabla direktorijuma. U osnovi funkcije **FormTree** je korišćenje DOS funkcije **findfirst** i **findnext**. Funkcije daju listu datoteka na osnovu zadate maske imena i atributa datoteka u nekom direktorijumu. Da bi se formiralo kompletno stablo za ceo disk, nije dovoljan jednostavan ciklus kao za datoteke jednog direktorijuma. Više rešenja ovog problema bilo je dosad opisano u *Računarima*. Zarad jednostavnosti (i lepote) mi smo se opredelili za rekurzivni algoritam. Poznata osobina rekurzije na računaru – proždrljivost prema memoriji – ovde nam nije mnogo smetala, jer uobičajena dubina stabla direktorijuma nikada nije tako velika.

Funkcija **FormTree** radi tako što za zadatu stazu (koja na nultom nivou uvek predstavlja root diska) izlista sve poddirektorije, i za svaki od njih pozove samu sebe. U tom novom pozivu funkcije ponavlja postupak sa novom stazom (poddirektorijum prethodnog direktorijuma). Svaki sledeći čvor račvanja stabla izaziva novi rekurzivni poziv funkcije.

Nastavljamo rubriku o Turbo Pascalu koju smo pokrenuli u prošlom broju. Ovog meseca ćemo preći dve teme: formiranje programskih jedinica (Units) i računanja vremena i datuma. U skladu sa klišeom kojeg ćemo se držati i ubuduće, kratke i (nadamo se) jezgrovite lekcije potkrepljujemo sa jednostavnim demo programima.

UNITS

Počev od verzije 4.0 pa na dalje, *Turbo Pascal* omogućava programeru da sačini sopstvene zbirke procedura i funkcija (**Units**) koje nakon prevođenja postaju praktično ravnopravne sa ugrađenim Borlandovim jedinicama **Crt**, **Dos**, **Printer**, **Graph** i dr. Njihovom formiranju se uglavnom pribegava u sledeća dva slučaja: prvo, kada korisnik želi da pojača TP biblioteku novih komandi, na primer za manipulisanje stringovima, i drugo, kada se ukaže potreba da se razvoj nekog glomaznog paketa podeli na nekoliko logičkih celina kao što su, recimo, glavni modul, univerzalne rutine i korisnički interfejs.

Za oba slučaja važi ista sintaksa. U prvom redu se umesto rezervisane reči *program* stavlja *unit* ispred deklaracionog dela listinga umesto će se oznaka *interface*, a ispred izvršnog dela – *implementation*. Svi tipovi i sve konstante i varijable koje se nalaze između njih vidljive su kako procedurama i funkcijama u toj jedinici, tako i programu koji ju je uključio svojom *uses* klauzulom.

Na samom kraju *interface* dela obavezno je dati spisak funkcija i procedura koje će se pozivati iz van programske jedinice. Taj spisak dosta podseća na *forward* deklarisanje, s tim da se ta rezervisana reč ne piše jer je i bez nje omogućeno pozivanje sa bilo kojeg mesta u izvršnom delu listinga. Procedure i funkcije koje nisu tu navedene, a postoje u biblioteci/modulu mogu se koristiti samo interno.

Uobičajeno je da programska jedinica nema glavni izvršni blok. Ponekad je pak potrebno izvesti inicijalizaciju u jedinici, kreiranjem neke labele ili sloga koji će biti potreban komandama pri kasnijem radu. U tom slučaju

Listing 1 DATETIME.PAS

```
{
        Primer biblioteke korisnickih komandi;
        Miroslav [illegible]     (c) September 1991
}
INTERFACE

uses   Dos;

const
MNames : array [1..12] of string = ( 'Januar', 'Februar',
         'Mart', 'April', 'Maj', 'Juni', 'Juli', 'Avgust',
         'Septembar', 'Oktobar', 'Novembar', 'Decembar' );

MDays  : array [1..12] of byte = (31,29,31,30,31,30,31,31,30,31,30,31);

DNames : array [0..6] of string = ( 'Nedelja', 'Ponedeljak', 'Utorak',
                         'Sreda', 'Cetvrtak', 'Petak', 'Subota' );

type   Date = record  Year, Month, Day, DayOfWeek : word  end;
       Time = record  Hour, Minute, Second, sec100 : word  end;

var    ElapTime   : Time;
       DayOfWeek  : word;

function  GetTimeStr   (T1 : Time) : string;
function  GetDateStr   (D1 : Date) : string;
function  GetDayOfWeek (D1 : Date) : string;
procedure GetElapTime  (T1,T2 : Time);


IMPLEMENTATION

function GetTimeStr(T1:Time) : string;
var    s1,s2,s3 : string;
begin
  Str(T1.Hour,  s1); if Length(s1)=1 then s1:='0'+s1;
  Str(T1.Minute,s2); if Length(s2)=1 then s2:='0'+s2;
  Str(T1.Second,s3); if Length(s3)=1 then s3:='0'+s3;
  GetTimeStr:=s1 + ':' + s2 + ':' + s3;
end;

function GetDateStr(D1:Date) : string;
var    s1,s2,s3 : string;
begin
  Str(D1.Day, s1); if Length(s1)=1 then s1:='0'+s1;
  s2:=MNames[D1.Month];
  Str(D1.Year,s3);
  GetDateStr:=s1 + [illegible] + s2 + [illegible] + s3 + [illegible];
end;

function GetDayOfWeek(D1:Date) : string;
var    D2 : Date;
begin
  GetDate(D2.Year,D2.Month,D2.Day,D2.DayOfWeek);
  SetDate(D1.Year,D1.Month,D1.Day);
  GetDate(D1.Year,D1.Month,D1.Day,DayOfWeek);
  GetDayOfWeek:=DNames[DayOfWeek];
  SetDate(D2.Year,D2.Month,D2.Day);
end;

procedure GetElapTime(T1,T2:Time);
var    Overflow,s1,s2,s3 : shortint;
begin
  s1:=T2.Second - T1.Second;
  if s1<0 then begin s1:=60+s1; Overflow:=1; end else Overflow:=0;
  s2:=T2.Minute - T1.Minute - Overflow;
  if s2<0 then begin s2:=60+s2; Overflow:=1; end else Overflow:=0;
  s3:=T2.Hour - T1.Hour - Overflow;
  if s3<0 then s3:=24+s3;
  ElapTime.Hour:=s3; ElapTime.Minute:=s2; ElapTime.Second:=s1;
end;

end  {unit}
```

Biblioteka komandi za rad sa datumima i vremenom

Listing 2 CLOCK.PAS

```
{
        Primer pozivanja korisnickove biblioteke komandi;
        Miroslav [illegible]          (c) September 1991
}
uses Crt, Graph, Dos, DateTime;

var  Gd, Gm                        : integer;
     Sx, Sy, Dux, Vis              : word;
     Xasp, Yasp, r1, r2, r3, n     : word;
     Ugao1, Ugao2, Ugao3, Odnos    : real;
     Prompt1, Prompt2, Prompt3     : string;
     Vreme1, Vreme2, Vreme3        : Time;
     Datum                         : Date;

procedure IscrtajKazaljku(Ugao:real; Rad1,Rad2:integer; Boja:word);
var  x1,x2,x3,x4 : integer;
     y1,y2,y3,y4 : integer;
begin
  x1:=Dux+round(Sin(Ugao+(PI/1))*Rad2/Odnos);
  y1:=Vis-round(Cos(Ugao+(PI/2))*Rad2);
  x2:=Dux+round(Sin(Ugao)*Rad1/Odnos);
  y2:=Vis-round(Cos(Ugao)*Rad1);
  x3:=Dux+round(Sin(Ugao-(PI/1))*Rad2/Odnos);
  y3:=Vis-round(Cos(Ugao-(PI/1))*Rad2);
  x4:=Dux+round(Sin(Ugao-PI)*Rad1/3/Odnos);
  y4:=Vis-round(Cos(Ugao-PI)*Rad1/3);
  SetColor(Boja); MoveTo(x1, y1);
  LineTo(x2, y2); LineTo(x3, y3);
  LineTo(x4, y4); LineTo(x1, y1);
end;

begin
  DetectGraph(Gd,Gm);
  InitGraph(Gd,Gm,'[illegible]');
  Dux:=GetMaxX div 2;
  Vis:=GetMaxY div 2;
  GetAspectRatio(Xasp,Yasp);
  Odnos:=Xasp/Yasp;
  SetTextStyle(GothicFont,HorizDir,1);
  SetTextJustify(CenterText,CenterText);
  Ugao1:=PI/6;
  SetFillStyle(SolidFill,0);
  for n:=1 to 12 do begin
    Sx:=Dux+round(Sin(Ugao1)*Vis*0.8/Odnos);
    Sy:=Vis-round(Cos(Ugao1)*Vis*0.8);
    Str(n,Prompt1);
    SetColor(8); PieSlice(Sx,Sy,0,360,18);
    if Gd=HercMono then SetColor(7) else SetColor(17);
    SetTextXY(Sx,Sy,Prompt1);
    Ugao1:=PI/6+Ugao1
  end;
  SetFillStyle(1,0);
  r1:=round(Vis*0.7);
  r2:=round(Vis*0.75);
  r3:=round(Vis*0.55);
  GetDate(Datum.Year,Datum.Month,Datum.Day,Datum.DayOfWeek);
  GetTime(Vreme3.Hour,Vreme3.Minute,Vreme3.Second,Vreme3.Sec100);
  repeat
    GetTime(Vreme1.Hour,Vreme1.Minute,Vreme1.Second,Vreme1.sec100);
    repeat
      GetTime(Vreme2.Hour,Vreme2.Minute,Vreme2.Second,Vreme2.sec100);
    until Vreme1.Second<>Vreme2.Second;
    SetColor(0);
    GetElapTime(Vreme3,Vreme2);
    if Vreme1.Hour<>Vreme2.Hour then begin
      GetDate(Datum.Year,Datum.Month,Datum.Day,Datum.DayOfWeek);
      Bar(Dux-135,round(Vis*1.35-12),Dux+135,round(Vis*1.15+12));
    end;
    Prompt2:=GetDayOfWeek(Datum) + [illegible] + GetDateStr(Datum);
    Prompt1:=GetTimeStr(Vreme2) + [illegible] + GetTimeStr(ElapTime);
    SetTextXY(Dux,round(Vis*1.15),Prompt2);
    Bar(Dux-105,round(Vis*1.5-17),Dux+105,round(Vis*1.5+18));
    SetTextXY(Dux,round(Vis*1.5),Prompt3);
    Ugao1:=PI/30*Vreme1.Second;
    SetColor(0);
    MoveTo(Dux,Vis);
    LineTo(Dux+round(Sin(Ugao1)*r1/Odnos),Vis-round(Cos(Ugao1)*r1));
    Ugao1:=PI/30*Vreme2.Second;
    SetColor(15);
    MoveTo(Dux,Vis);
    LineTo(Dux+round(Sin(Ugao1)*r1/Odnos),Vis-round(Cos(Ugao1)*r1));
    if Vreme1.Minute <> Vreme2.Minute then begin
      Ugao2:=PI/30*Vreme1.Minute;
      Ugao3:=(PI/6*Vreme1.Hour) + (PI/360*Vreme1.Minute);
      IscrtajKazaljku(Ugao2,r2,8,15);
      IscrtajKazaljku(Ugao3,r3,12,15);
    end
    else begin
      IscrtajKazaljku(Ugao2,r2,8,0);
      IscrtajKazaljku(Ugao3,r3,12,0);
      Ugao1:=PI/30*Vreme2.Minute;
      Ugao3:=(PI/6*Vreme2.Hour) + (PI/360*Vreme2.Minute);
      IscrtajKazaljku(Ugao1,r1,8,15);
      IscrtajKazaljku(Ugao3,r3,12,15);
    end;
  until KeyPressed;
  CloseGraph;
end.
```

Primer pozivanja biblioteke **DATETIME**



se formira i glavni blok koji se, da zamislimo, izvršava pri samoj uses klauzuli!

Programske jedinice mogu da pozivaju i jedna drugu. Potrebno je u svakoj deklarisati korišćenje one druge putem *uses*. Tada prva može da pozove drugu, druga prvu, prva opet drugu i tako ukrug. Pogodno za stvaranje opšte zbrke i prepunjavanje steka. U praksi se uobičavaju hijerarhijski odnosi između jedinica bez međusobnog pozivanja.

Konačno, moduli aplikacija čiji izvršni kod prelazi dozvoljeni limit od 65535 bajtova se preklapaju tako da se u taj deo memorije prenose samo moduli neophodni za izvršavanje programa. Preklapanjima (**overlays**) se upravlja specijalnim setom direktiva prevodiocu o čemu ćemo pisati u jednom posebnom izdanju ove rubrike.

DATE&TIME

Za razliku od većine drugih dobrih i popularnih kompajlera (prelistajte samo uputstva za *Clipper*), *Turbo Pascal* je više nego 'tanak' sa instrukcijama za rad sa sistemskim vremenom i datumom. Na raspolaganju su nam samo dve najosnovnije procedure – **GetTime** i **GetDate** – dok se za svu nadgradnju moramo sami pobrinuti pišući rutine koje u ostalim jezicima već dobijamo gotove!

Kao primer male biblioteke novih komandi za rad sa datumom i vremenom donosimo listing 1.

Funkcija GetTimeStr (T1 : Time) : string;

Pretvara prosleđeni slog tipa **Time** u string dužine osam karaktera. Ovu funkciju znači treba koristiti pri svakom ispisu vremena na ekran.

Funkcija GetDateStr (D1 : Date) : string;

Transformiše slog tipa **Date** u string koji čine dan u mesecu, naziv meseca na srpskohrvatskom i godina. String je ovde usled prisustva naziva meseci promenljive dužine: najmanje 12 (maj) a najviše 18 znakova (septembar).

Funkcija GetDayofWeek (D1 : Date) : string;

Vraća ime dana u nedelji prosleđenog datuma. Ova funkcija u stvari zloupotrebljava postojeće **SetDate** i **GetDate** procedure tako što prvo sačuva tekući datum u jednom pomoćnom slogu, zatim ga zameni sa traženim datumom, dobije kod za dan u nedelji i naposletku vrati tekući datum. Kako se sistemski datum i vreme srećom čuvaju u posebnom RAM-u koji se osvežava i dok je računar isključen, a ne na disku, to je ovakav pristup sasvim upotrebljiv. Ostaje međutim ozbiljno ograničenje da datum argument funkcije ne može da bude izvan opsega 01.01.1980 – 31.12.2099 – bolje rešenje bi, prema tome, bila nezavisna funkcija koja bi kao prvo izračunavala broj proteklih dana između prosleđenog datuma i nekog za koga znamo u koji je dan padao, a potom koristeći *mod* operacije nalazila dan u nedelji. Ovakva jedna funkcija nalazila se u prvobitnoj verziji biblioteke DATETIME ali je pre svega zbog dužine, a zatim i zato što se nije uklapala u demo-program, izbačena, zajedno sa još par atraktivnih ali *king size* rutina koje nam mogu, pomenimo, naći broj sekundi proteklih do isteka ovog veka ili datum kojeg ističe garancija na vaš novi kompjuter. Bude li interesovanja da se i one objave, mi ćemo to rado učiniti u nekom od narednih brojeva.

Procedura GetElapTime (T1,T2 : Time);

Nakon poziva ove procedure globalna promenljiva **ElapTime** će sadržavati razliku u vremenu između T1 i T2. Zbog čega je ovo procedura a ne funkcija? Zato što je **ElapTime** slog od četiri reči (tip **Time**) a funkcije mogu vraćati isključivo proste tipove podataka – mana koja je potpuno prevaziđena u objektnim jezicima, a delimice u i poslednje dve verzije *Turbo Pascala*.

Obratite pažnju da nijedna od ovih komandi ne proverava ispravnost poslatih argumenata: datumi kao 30.02.1985. ili 31.14.1990. te vremena poput 54:23:17 jasno nemaju smisla. Komande će doduše vraćati neke vrednosti, no one će biti jednako besmislene. Ako nameravate da koristite ove rutine u svojim projektima, obavezno unutar biblioteke dodajte i dve interne funkcije za kontrolu ispravnosti argumenata. Njih svaka od ova četiri komande treba da konsultuju pred početak sopstvenog rada.

TAČNO JE...

Kao primer upotreba svih komandi iz nove biblioteke dat je na drugom listingu program CLOCK, inspirisan istoimenom mini aplikacijom iz Windowsa. Jeste da je malo podugačak za jedan demonstracioni program, ali ako ga pažljivo prostudirate možete uočiti par korisnih trikova pri prikazivanju časovnika – pomenimo samo redosled iscrtavanja kazaljki i obnavljanja ispisa vremena i datuma!

Na kraju valja reći nešto i o konvencijama u pisanju programa, preciznije o nazivima identifikatora: sad je posebno zgodna prilika pa je ne treba propustiti. Naime, radi se o sledećem: pokazalo se kao dobra praksa da se koriste nazivi na srpskohrvatskom kako bi se i na prvi pogled mogli lako razlikovati od ugrađenih identifikatora i rezervisanih reči. Nijedan programer u paskalu, ma koliko iskusan, ne zna i ne mora da zna napamet imena svih sistemskih tipova, konstanti, varijabli, funkcija, procedura i ostalog, pa je zato poželjno da njihova nazive pri debagiranju što lakše razlikuje od sopstvenih. Navika da se oni pišu na našem ima naročito opravdanja kod dugih zamršenih programa na kojima se radi i po više meseci – o slučaju kada u njihovom pisanju učestvuje više programera da i ne govorimo.

Sad jasno sledi opaska: zašto ovaj koji to tako preporučuje komandama iz vlastite biblioteke daje engleske nazive!? Odgovor je naravno više nego očigledan – nakon prevođenja te komande postaju sastavni i ravnopravni deo jezika. Nije potrebno da programer, kada traži greške u svom programu, obraća pažnju na njih jer se još ranije uverio u njihovu ispravnost!

U sledećem broju se igramo sa sprajtovima.

<<< EOF >>>

OSMORICA VELIČANSTVENIH

Nastavak sa strane 18

sike između ova dva modela.

Odlična mašina, čak i u kontekstu cene – brza ploča i brz i veliki disk. Jaka preporuka.

GAMA 486DX-33

Procesor, takt: Intel 80486DX, 33 MHz
Memorija: 8 MB
Tvrdi disk: Western Digital 210 MB
Video podsistem: VGA/EVGA/SVGA
Ocena (M) 66.6% (EI) 60.3% (P) 373%

I za kraj, trenutno najjača mašina koju „Gama" nudi. Naravno, radi se o računaru sa procesorom Intel 80486DX na 33 MHz, sa 256 KB drugostepene keš memorije. Matična ploča ima format „baby AT" ploče, sa ukupno osam utičnica za proširenja. Od memorije na nju možete staviti osam standardnih SIM modula, ali postoje i dva posebna postolja za do sada neviđene dvostruke (duple) SIM module. Koristeći sva mesta, memoriju možete proširiti do 32 MB, odnosno do duplo veće granice od nekakvog prihvaćenog standarda.

Druga specifičnost ove ploče se ogleda u njenom BIOS-u, koji potiče od do sada malo poznate firme „MR". Bez obzira na to, mirne duše možemo reći da je ovo do sada najbolji BIOS koji smo ikada imali prilike da vidimo, bar što se podešavanja tiče. Za razliku od AMI BIOS-a, koji nudi pristup raznim registrima od kojih neke znaju valjda samo dva-tri inženjera Intel-a, ovaj BIOS nudi logično prikazane i organizovane parametre rada posredstvom menija, za koje je potrebno sasvim malo znanja tehnike.

Dok smo na temi BIOS-a, ova ploča je njegovim posredstvom izuzetno programabilna: recimo, sabirnicu možete podesiti na radne taktove od 5.6, 8.3, 11.1 i 13.3 MHz. Sem najmanjeg, probali smo sve radne taktove i sve je besprekorno radilo. Međutim, podvukli bismo jedan detalj ovde, za koji će se mnogi „zalepiti": iako sa povećanjem radnog takta brzina videa raste, kao i nominalna brzina prenosa sa diska (vidi grafikon), realno ubrzanje je samo prelazak sa 8.3 na 11.1 MHz. U narednom koraku (tj. na 13.3 MHz), stvarna brzina diska, prema testovima sa programima, počinje da opada, pa se suočavate sa kompromisom – još veća brzina videa po cenu manje brzine diska.

U merenjima smo koristili brzinu sabirnice od 8.33 MHz jer se radi o relativnom standardu sa jedne i sa druge strane: to je brzina na kojoj se očekuje da bi sve što stavite u računar moralo da radi – 11.1 ili 13.3 MHz je već podosta iznad standarda. Međutim, ako ne planirate da dodajete druge stvari, preporučili bismo vam brzinu sabirnice od 11.1 MHz kao najsrećnije rešenje: odnosno malo povećanje brzine u svim domenima.

Kao što se iz prikaza performansi vidi, ovaj računar je glatko zauzeo prvo mesto po performansama: jednostavno, ovo je najbrže što smo ikada imali prilike da probamo. Indeks performansi u mašinskim testovima je odličnih 2.72 (ili, sa sabirnicom na 13.3 MHz, 2.91), a u programskim testovima 3.73 (nema promene na 13.3 MHz: video je brži, ali je disk sporiji pa se potiru). Zaista je zadovoljstvo videti ovu mašinu kako radi sa CAD-om.

Interesantno je primetiti da bolji rezultati ovog računara u odnosu na „Most" 486-33 ne potiču isključivo od diska, već u maltene svim testovima, uprkos istoj osnovnoj konfiguraciji.

Najzad, odnos cene, performansi i kvaliteta je veoma dobar: ako vam treba mašina brza kao đavo, ovaj model stavite na vrh spiska.

Čist slučaj za „R/Izbor".

400 [BASE CLIPPER, PC] Za jubilarni „Bajt lične prirode" mala poslastica – program u *Clipper*-u (slika 1) koji će proizvoljni numerički iznos ispisati slovima (npr. broj 3950 se transformiše u „tri hiljade devet stotina pedeset dinara"). Koliko je to potrebno za razne komercijalne programe nama ni potrebe objašnjavati.

Ovaj korisni potprogram napisali su (i za potrebe Sezama proglasili javno dostupnim) programeri zrenjaninskog preduzeća „Mihajlo Petrović Alas" 023 60 020. Prilog je na Sezam poslao **Dragan Nedeljković** (*ndragan*). Program slične namene koji ovde ne objavljujemo (ali ga, naravno, možete preuzeti sa Sezama) poslao je i **Ivan Todorović** (*ivantod*).

401 [MASM, PC] Već par puta govorili smo o IDE (AT bus) diskovima i preporučivali da nikako ne pokušavate njihovo fizičko (*low level*) formatiranje. Evo i zbog čega – neki od diskova ovoga tipa koriste specijalne analogne servo signale (*servos* u tehničkim dokumentacijama) koji omogućavaju glavi diska da precizno locira sektore i cilindre na disku na osnovu koordinata. IDE kontroleri su uglavnom jednostavne naprave u koje se ne ugrađuje specijalna elektronika neophodna za kreiranje ovih signala. Formatiranje takvog diska može umanjiti njegovu pouzdanost ili čak totalno onemogućiti njegov rad, posle čega je potrebno servisiranje.

Ove servo signale, međutim, ne koriste svi IDE diskovi, pa se neki od njih mogu fizički formatirati premda za tim nikada ne bi trebalo da postoji potreba. Međutim, ako ne posedujete preciznu tehničku dokumentaciju proizvođača u kojoj je ovaj problem detaljno opisan, **nikako** ne pokušavajte *low level* formatiranje IDE diska – lako je moguće da ćete ga time pokvariti. [Preuzeto iz PC Magazine-a]

402 [MASM, PC] Za razliku od svojih IDE naslednika, „starim" MFM i RLL diskovima *low level* formatiranje ne samo da ne škodi nego i prija. Disk tokom vremena doživljava određene mikro-promene pošto se njegove površine usled toplote i neprekidnog obrtanja deformišu. To vremenom može dovesti do pojave novih loših sektora i ugroziti podatke. Zato mnogi proizvođači preporučuju da se u proseku jednom godišnje disk *low level* formatira, čime se obezbeđuje njegov dug i pouzdan rad.

Obzirom da se formatiranjem detaljno i nepovratno uništavaju svi podaci na disku, prethodno treba napraviti *backup* i proveriti ispravnost disketa ili traka na koje su podaci preneseni.

403 [ASEM, PC] Kome se nije desilo da mu se program blokira? Prema teoriji, tada treba pritisnuti **Ctrl Break**, ali u praksi to ne radi baš uvek pa se kompjuter mora resetovati. Osnovno objašnjenje za to je da je startovani program upao u neku petlju koja uopšte ne poziva DOS, pa se **Ctrl Break** ni ne testira.

Rezidentni program sa slike 2 preuzima INT 9h i reaguje kako na **Ctrl Break** tako i (još bolje) na **Ctrl Alt 4** (broj 4 iz gornjeg reda, a ne sa numeričke tastature. Trebalo bi da može da prekine „svaki" program, osim u slučaju da je DOS totalno „ugušen" ili da je došlo do problema u interapt sistemu PC-ja. [Preuzeto sa Bitnet-a, CCUC@UMCVMB, na Sezam preneo Božidar Radunović (radunovic)]

404 [OS MSDOS, PC] Evo nekoliko zanimljivih zaključaka o atributima datoteka, naravno postavljanim na neuobičajene načine. Prvi fajl u *root* direktorijumu na koji DOS naleti a koji ima setovan VOL atribut biva prihvaćen kao ime diska, dok se eventualni ostali fajlovi sa VOL atributom (koji legalno ne bi ni trebalo da mogu nastati) ne konstatuju. Komandom LABEL ne može se promeniti ime diska ako „kritična" datoteka sa VOL atributom ima i setovan neki od DIR, R/O, SYS ili HID atributa – ovo je zanimljiv način za zaštitu diska od neželjenog formatiranja.

Ako neki fajl ima VOL atribut, ne pojavljuje se u ispisu koji proizvodi komanda DIR. Ako je taj fajl slučajno direktorijum (DIR atribut mu je setovan), nemoguće je ući u njega (ne postoji) pa ga DOS ne može ni obrisati. I to ne samo DOS nego ni *Norton Commander*. Ako neki od programa ne obraća pažnju na postojanje VOL atributa na datotekama na kojima je DIR atribut setovan (traži samo DIR), može se ukinuti DIR a staviti R/O i ime diska će biti nevidljivo i nepromenljivo komandom LABEL. CHKDSK u takvom slučaju prijavljuje izgubljene klastere ali se, baš kao ni *Norton Disk Doctor*, neće zbuniti nego će te klastere povezati u datoteke. [Preuzeto sa Sezama, autor **Đorđe Ratnović** (*dj ratnovic*)]

405 [ZAST, PC] Istočnom (a uskoro svakako i zapadnom) Evropom širi se koncepcijski novi i veoma opasni virus nazvan DIR-2. Za razliku od dosadašnjih virusa koji su inficirali datoteke ili *boot* sektor diska, DIR-2 primenjuje sasvim novi metod. Prilikom infekcije program se upiše u slobodan prostor na kraju diska, a onda modifikuje podatke u direktorijumima tako da svaki izvršni program navodno počinje na mestu gde se virus nalazi. Podaci o pravom početku programa se upisuju u neke od nekorišćenih bajtova zone direktorijuma. Posle toga svako startovanje bilo kog programa zapravo pokreće virus a on, pošto uradi šta ima, sam aktivira traženi program.

Rezultati su vrlo neprijatni – pre svega, same datoteke se ne menjaju pa svaki od postojećih anti virus programa može samo da primeti da je „sve u redu". Drugo, datoteke se mogu preneti na diskete i može se napraviti *backup* ali se zajedno sa njima prenosi i virus. Treće, ako se sistem startuje sa „čiste" DOS diskete, ni jedan izvršni program sa diska ne može da se prekopira na diskete niti se može napraviti *backup*. Neoprezna upotreba CHKDSK /F će u takvom slučaju uništiti veći deo datoteka na disku.

Virus se, prema podacima sa *Internet*-a, širi velikom brzinom po Bugarskoj, Poljskoj, Jugoslaviji i Mađarskoj. *Veselin Bončev* iz Bugarske javlja da virus poslednjih nedelja pravi neverovatan haos na diskovima mnogih njegovih zemljaka.

Za sada jedino efikasno oruđe za otkrivanje DIR-2 virusa je novi SCANV84.ZIP firme *McAfee Associates* (telefon USA (408) 988-3832, BBS USA (408) 988 4004) – možete ga, naravno, preuzeti i sa Sezama. Metod za lečenje još nije pronađen – treba spasti što se spasti može (izvršne programe ne treba kopirati jer će se zapravo kopirati virus) i onda formatirati disk. [Preuzeto sa Internet-a, autor **John McAfee**, *mcafee@netcom.com*]

406 [COMM, PC] Evo nekoliko iskustava o vrednostima koje treba upisati u pojedine registre *Hayes* kompatibilnih modema da bi se oni što bolje prilagodili našim telefonskim linijama. Vrednost registra se postavlja komandom **ATSx**=y, gde je *x* oznaka registra a *y* vrednost koja se upisuje u njega. Komande ovog tipa treba ugraditi u inicijalizacionu sekvencu modema.

Registar S7 se obično postavlja na 255, što je verovatno previše jer je u samom komunikacionom programu timeout podešen na nekih 30-tak sekundi. Postavljanjem S7 na manju vrednost (S7=10 je zgodno za gradske pozive) obezbeđuje se poruka NO CARRIER kod poziva koji zvone u prazno, tj. kada se veza ne uspostavi posle 2-3 zvuka. Tih 10-15 sekundi i nije mala vrednost kao što izgleda, jer se vreme broji posle zadnjeg otkucanog broja. Pošto neki komunikacioni programi sami naknadno postave S7, treba umesto ATDP prefiksa u konfiguraciji postaviti ATS7=10DP – u protivnom prethodno ATS7=10 gubi smisao.

Registar S9 po *default*-u ima vrednost 6 što je za naše prilike mnogo. Treba postaviti S9=1. U tom slučaju BUSY signal će biti detektovan skoro sa sigurnošću (za *ViVa* modeme još uz opciju &P1 – *make/break ratio 33%/67%*). Tako konfigurisan, modem bi trebao da otkriva BUSY posle 3-5 zvukova. Ukoliko ga i ne otkrije, naletece na NO CARRIER (iz S7), pa će postati potpuno nepotrebno mehanički prekidati vezu radi što bržeg dobijanja veze.

Registar S6 je obično postavljen na 3, ali se može postaviti i na 2, čime se štedi jedan sekund po pozivu. Sekund zaista nije mnogo, ali sekund po sekund... [Preuzeto sa Sezama, autor **Aleksandar Stojadinović** (*stal*)]

407 [MREZ NOVELL, PC] Jedan od zaštitnih mehanizama *Novell*-a omogućava „povraćaj" svake obrisane datoteke. I to ne nalik na onu nepouzdanu operaciju koju nude programi kao što su *Norton Utilities* ili *PC Tools* – datoteka se povraća komandom SALVAGE i to uvek 100% uspešno, naravno, pod pretpostavkom da se korisnik nije od trenutka brisanja odjavljivao sa mreže. Sve je to lepo u interaktivnom radu, ali kada mreža radi automatski bez ljudskog nadzora, pretvara se u svoju suprotnost. Uzmimo samo Sezam – njegovi nodovi rade neprekidno 24 časa dnevno, sedam dana u nedelji. Bez neke preke potrebe ih ne resetujemo. U takvim uslovima svaka obrisana datoteka bez ikakve potrebe ostaje na disku, što smanjuje raspoloživ prostor – znalo se desiti da posle više dana rada resetujemo sve nodove i na disku se oslobodi nekoliko megabajta prostora!

Na svu sreću, *Novell* je mislio o ovom problemu, ali tek u verziji za 386 – dovoljno je da se u datoteku AUTOEXEC.NCF ugradi **SET Immediate Purge Of Deleted Files**=ON. [Preuzeto sa Sezama, autor **Ivan Karaei** (*ikarasi*)]

408 [RAC; DZEP] Jeste li znali da HP48SX ima hardverski *reset*? Treba samo skinuti gornju levu gumenu nožicu i nakratko iglom dotaći rupicu pored koje piše R. Ovome, naravno, treba pribeći tek ako je standardni *reset* (ON A F) iz nekog razloga nemoguć. [Preuzeto sa Sezama, autor **Vladimir Ilić** (*vilic*)]

409 [IGRE, PC] Igrate li još *Prince of Persia*? Ako tek počinjete, probajte da umesto PRINCE otkucate PRINCE MEGAHIT. Posle sa Shift L možete prelaziti na sledeće nivoe bez ograničenja.

Pri pozivu programa mogu se navesti još neki nedokumentovani argumenti koje možete videti ako programom NU (ili nekim sličnim) pregledate PRINCE.EXE. Ako neko otkrije smisao tih argumenata, rado ćemo objaviti prilog. [Preuzeto sa Sezama, autor **Kiro Velkovski** (*kvelkovski*)]

410 [KOMS, PC] Radite li sa Borlandovim *Turbo Debugger*-om? I jeste li pokušali da, na primer, postavite prekidnu tačku na 123D? Ako jeste, našli ste na problem – jednostavno ne funkcioniše, i to tako uporno da će vas iz kože isterati. Ovo je još jedna od onih stvari koje su kad ih znate očigledne, ali dok se ne setite... slovo D na kraju nije interpretirano kao hex cifra nego kao postfiks koji označava da se radi

Listing 1

```
* Proglašeno za javni program 11 11 1111
* prilikom upotrebe imate obavezu da nas negde spomenete u dokumentaciji
* Od "Mihajlo Petrović Alas" (MPA) Ivanjevin
* 111/11-111  lako/nadam 35 111
* program ne radi sa negativnim brojevima
* upotreba: Cvar = slovima(Nvar)
* izmedju reči koje program generiše nema razmaka  jer sam lakše snalaženje
* dobili od odvajanja knjigovođa i ...
*
FUNCTION slovima
PARAMETER broj
PRIVATE sbroj trio,ce,cd,cj,i res, deli  dec
IF broj=0
   RETURN  nula
ENDIF
DECLARE imebr[9]
imebr[2]= jedan
imebr[3]= dva
imebr[4]= tri
imebr[5]= četiri
imebr[6]= pet
imebr[7]= šest
imebr[8]= sedam
imebr[9]= osam
imebr[9]= devet
STORE    TO res
celi=Int(broj)
dec = (broj - celi)*100
STORE STR(celi 15) TO sbroj
STORE 1 TO i
DO WHILE i<15
   STORE SUBSTR(sbroj i,3) TO trio
   IF trio[      AND (trio[     OR i=1)
      STORE VAL(SUBSTR(trio 1 1)) TO ce
      STORE VAL(SUBSTR(trio 2 1)) TO cd
      STORE VAL(SUBSTR(trio 3 1)) TO cj
      DO CASE
         CASE ce=1
              res=res+ sto
         CASE ce=2
              res=res+ dve
         CASE ce>2
              res=res+imebr[ce]
      ENDCASE
      DO CASE
         CASE ce>4
              res=res+ stotina
         CASE ce>1
              res=res+ stotine
      ENDCASE
      DO CASE
         CASE cd=1
              res=res+ deset
         CASE cd=4
              res=res+ četr
         CASE cd>1
              res=res+imebr[cd]
         CASE cd=1
              DO CASE
                 CASE cj=1
                      res=res+ danaest
                 CASE cj=1
                      res=res+ jeda
                 CASE cj=1
                      res=res+ dela
                 OTHER
                      res=res+imebr[cj]
              ENDCASE
              IF cj>0
                 res=res+ naest
              ENDIF
      ENDCASE
      IF cd>1
         res=res+ deset
      ENDIF
      IF cd[1 AND cj[( AND  i=13 OR VAL(trio)[4]
        IF cj=2 AND i[13 AND i[7
           res=res+ dve
        ELSE
           res=res+imebr[cj]
        ENDIF
      ENDIF
      DO CASE
         CASE i=1 OR i=11
              res=res+ bilijad
              DO CASE
                 CASE VAL(trio)=1
                      res=res+ a
                 CASE cj>4 OR cj=1
                      res=res+ a
                 CASE cj>1
                      res=res+ e
              ENDCASE
         CASE i=4
              res=res+ milijard
              DO CASE
                 CASE VAL(trio)=1
                      res=res+ a
                 CASE cj>4 OR cj=0
                      res=res+ i
                 CASE cj>1
                      res=res+ e
              ENDCASE
         CASE i=1
              res=res+ milion
              IF cj[1
                 res=res+ a
              ENDIF
      ENDCASE
   ENDIF
   i=i+3
ENDDO
cdec=str(dec 2)
IF asc(cdec)=32
   cdec= 0 +str(dec,1)
endif
slov=res+   +cdec+ /100
RETURN slov
```

Program za ispisivanje slovima proizvoljnog numeričkog zapisa (na primer: 37 kao trideset sedam)

Listing 2

```
;Resident program to enable ctrl-break even if an DOS call is made
;This is done by doing a dummy DOS call when break key is let up
;DOES NOT WORK WITHOUT DOING 1 break on BUFFERS? IN DOS
;if ctrl break fails you may try ctrl-alt-4 (the 4 on the top row)
;which is somewhat more powerful  if you break out of basic this way
;the NEXT time BASIC is run  it will hang  requiring a reboot
;other programs (games?) may or may not respond  if they do respond
;they may hang up now
cseg      segment    para public  code
          org        100h
newbrk    proc far
          assume     cs:cseg ds:cseg
          jmp        install              ; this installation code done only once
oldint9   dd         ?                    ; storage for old int 9 vector
intloc    equ        9*4                  ; location of int 9
kbflag    equ        417h                 ; keyboard "hold down" status address
ctrl      equ        04                   ; bit in kbflag for ctrl
alt       equ        08                   ; bit in kbflag for alt
brkup     equ        0c6h                 ; scan code for release of scroll lock
ctrl4     equ        05h                  ; scan code for 4
newint9:                                  ; comes here from the hardware int 9
          sti                             ; restart interrupt
          push       ax                   ; save registers
          push       di
          push       es
          push       bx
          push       ds
          pushf
          mov        ax,0                 ; get ctrl and alt status into ah
          mov        es,ax
          mov        di,kbflag
          mov        ah,es:[di]
          test       ah,ctrl              ; test for ctrl held down
          jz         return               ; if ctrl is up  go to previous handler code
          in         al,60h               ; get scan code into al
          cmp        al,brkup             ; if not release of scroll lock or
          je         doit
          cmp        al,ctrl4             ; 4 key  go to previous code
          jne        return
          test       ah,alt               ; if alt not down with ctrl-4
          jz         return               ; go to previous code
doit:     mov        al,al                ; put a pulse in interrupt port
          in         al,61h               ; its some obscure reason
          mov        ah,al                ; copied from BIOS listing
          or         al,80h
          out        61h,al
          mov        al,ah
          out        61h,al
          cli                             ; stop interrupt
          mov        al,20h               ; clear keyboard
          out        20h,al
          sti                             ; restart interrupts
          cmp        al,ctrl4             ; test for ctrl-alt 4
          je         funix                ; if it is ctrl-alt-4 go to code for that
          mov        ah,19h               ; DOS func call for read default disk  This
          int        21h                  ; is a dummy call to when DOS look for break
          popf                            ; restore registers in case DOS break
          pop        ds                   ; didn't work
          pop        bx
          pop        es
          pop        di
          pop        ax
          iret                            ; this may never be used
funix:    mov        al,01h               ; error code for ctrl-break
          mov        ah,4ch               ; DOS func call to terminate program
          int        21h
          iret
return:   popf                            ; this code simply passes control to the
          pop        ds                   ; keyboard handler present before this
          pop        bx                   ; resident program was loaded
          pop        es
          pop        di
          pop        ax
          jmp        cs:[oldint9]         ; this jump doesn't change stack
install   mov        ax,0                 ; get old int 9 vector into
          mov        es,ax
          mov        di,intloc
          mov        ax,es:[di]           ; ax and bx
          mov        bx,es:[di+2]
          mov        si,offset oldint9    ; then put them into
          mov        [si],ax              ; location "oldint9" in this program
          mov        [si+2],bx
          mov        ax,0
          mov        es,ax
          mov        bx,ds
          cli                             ; stop interrupts
          mov        di,intloc            ; put pointer to this program into
          mov        ax,offset newint9    ; int 9 vector
          mov        es:[di],ax
          mov        es:[di+2],bx
          sti                             ; enable interrupts
          mov        dx,offset install    ; don't keep the install code
          int        27h                  ; terminate and stay resident
newbrk    endp
cseg      ends
end       newbrk
```

Rezidentni program za vađenje računara iz blokade

o broju u decimalnom sistemu. Otkucajte 123DH i sve će odjednom biti u redu. [Preuzeto sa Sezama, autor **Igor Milosavljević** (*igor mil*)]

411 [PJ PASCAL, PC] Kako u nekoj od objektno orijentisanih verzija Turbo Pascala snimiti objekte na disk? „Logično" *file of object* ne funkcioniše, pa se morate snalaziti „sakrivajući" objekte u neki slog (*record*) pa snimati njega. Ne postoji i „oficijelno" rešenje u fajlu OBJECTS.PAS nalaze se definicije i source klasa *Stream*, *DosStream* i *BuffStream* — ako pogledate OOPDEMOS.DOC videćete i kako se to koristi. [Preuzeto sa Sezama, autor **Ivan Stanisavljević** (*ivan s*)]

412 [OS WINDOWS, PC] Ukoliko koristite SCSI disk, već ste navikli na razne „varke" koje su neophodne da se obezbedi „saradnja" sa DOS-om: hard diska nema u SETUP-u, NDD vam stalno savetuje da ne nastavljate ako niste popili nekoliko aspirina i tome slično. Kada poželite da koristite *Windows 3.0*, pojavljuju se novi problemi koje srećom možete da rešite stavljajući *VirtualHDIrq=off* u SYSINI3.TXT. Podrazumevano stanje je naime *on*. Takođe, *permanent swap file* mehanizam ne radi sa SCSI diskovima, pa ga treba isključiti; *temporary swap file* bi trebalo da radi normalno. [Preuzeto sa Sezama, autor **Viktor Obuljan** (*viktor*)]

Priloge za ovu rubriku šaljite na adresu „Računari" (za „Bajtove lične prirode"), Bulevar vojvode Mišića 17, Beograd ili preko Sezama u vidu lične poruke uredniku (*mail write dejan*)

ZOVITE DA VIDITE

Kako vreme brzo prolazi – čini nam se kao da smo se pre samo nekoliko dana zagrevali za osnivanje sopstvenog BBS-a, a naš sistem radi već pune dve godine! I to radi uspešnije nego što smo i za trenutak smeli da se nadamo – protekla godina donela nam je mnoštvo novih članova, proširenje kapaciteta na 10 linija, više desetina hiljada poruka u konferencijama, megabajte i megabajte datoteka za *download*, povezivanje sa Tanjugovim servisima, i, naravno, mnogo zadovoljstva koje održavanje ovako dinamičnog sistema donosi. Zato ovaj tekst posvećujemo drugoj godini *Sezamovog* rada.

Ne možemo reći da se od dana kada smo počeli da radimo na koncipiranju *Sezama* i razvoju softvera koji će omogućiti doslednu realizaciju te koncepcije nismo nadali da će čitava akcija biti uspešna. Pa ipak, kriterijumi se s vremenom menjaju – kada smo prvi put razgovarali o ideji, činilo nam se da bi „uspeh" bio imati stotinak članova (tada naravno članarine nije bilo) i nekoliko poruka dnevno. Već prvi meseci rada pokazali su nam koliko su te procene bile skromne – *Sezam* se očito pojavio u savršenom trenutku, kada je naokolo postojala pristojna količina modema a malo profesionalno zasnovanih informacionih sistema i BBS-ova koji bi omogućili da se ti modemi iskoriste. Od velike pomoći je, naravno, bio i autoritet časopisa „Računari" i svih onih koji su radili na razvoju i održavanju *Sezama* – očigledno je da su čitaoci našeg časopisa verovali da će naš BBS biti nešto vredno truda! To je ujedno bio i dodatni podsticaj za mnoge čitaoce da se odluče za nabavku svog prvog modema – kada jednom pročitate izvode sa *Sezama* koje smo prvih meseci sistematski objavljivali u časopisu, prosto poželite da se i sami uključite u razgovor! To je bio onaj „zamajac" koji je tokom čitave prve godine rada pokretao *Sezam*, što je pokazalo da je njegova osnovna koncepcija korektna – sa pravom smo verovali da, u uslovima koji ovde vladaju, prvenstveni intres većine potencijalnih članova nije *download* softvera nego razmena informacija i mišljenja, i to kako računarskih tako i opštih.

NAŠ DAN D

Veliki broj članova i njihovo interesovanje za sistem ohrabrilo nas je da pred sam prvi rođendan *Sezama* krenemo korak dalje, tj. da sistem iz ipak „neobaveznog" BBS-a pretvorimo u profesionalni sistem za modemske komunikacije. Jedna od posledica te odluke moralo je da bude uvođenje pretplate što je, na-

Dejan Ristanović

ravno, bio veoma „opasan" trenutak – iako je bilo očigledno da se *Sezam* sviđa svima koji ga zovu, da li će dovoljan broj njih smatrati da su informacije koje na ovaj način dobija vredne godišnje članarine od 500 dinara (tada 70 DEM)? Prelazak na komercijalni rad se, naravno, nije smeo svesti na naplatu članarine – trebalo je korisnicima ponuditi nešto sasvim novo, a to je bio višelinijski rad. Tako smo avgusta 1990. u AT koji je skoro godinu dana verno „pokretao" *Sezam* ugradili mrežnu karticu i dodali dve stanice koje su u tom trenutku eksperimentalno radile pod dLINK-om. To je bila i prilika za do danas najradikalniju preradu *Sezamovog* softvera – osim koda neophodnog da bi se radilo u lokalnoj mreži (zaključavanje datoteka i slogova, komunikacija između nodova itd.) uveli smo i brojna izmene zasnovane na iskustvima koja smo stekli i predlozima korisnika. Uveli smo mehanizam PAD-a u koji mogu da se upisuju čak i datoteke, omogućili pisanje dužih poruka, razima korisnika, usavršili mogućnost da svaki korisnik napiše svoj AUTOEXEC... No najznačajniji novitet bio je *chat*, mogućnost da korisnici koji su istovremeno na vezi razgovaraju o temama koje ih zanimaju. Koncipirajući *chat* imali smo pre svega u vidu budućnost – odabrani mehanizam nije izgledao naročito atraktivno kada se radi na dve linije, ali smo ga „zamislili" na budućim *Sezamovim* resursima. I zaista – na današnjih 10 linija prvi sati svakog dana gotovo isključivo prolaze u *chat*u, a i u „normalnija" vremena se nađe puno korisnika (i korisnica) zainteresovanih za interaktivnu razmenu mišljenja.

Pošto smo mesec dana probno radili na dve linije i u tom intervalu se uverili da hardver i softver rade pouzdano (gotovo je neverovatno koliko smo malo problema imali sa softverom, pogotovu kada se uzme u obzir da se radi o veoma složenoj aplikaciji koja se može testirati jedino „na živo", u realnoj mreži koja je izložena svim mogućim i nemogućim situacijama koje korisnicima mogu pasti na pamet), bilo je vreme da se pređe na „pravu stvar", tj. komercijalni rad na tri linije. Oprema je montirana do 1. oktobra 1990, dok je 15. oktobar bio poslednji dan kada se *Sezamu* moglo besplatno pristupati – posle dosta razmišljanja, odlučili smo da neregistrovani korisnici nemaju nikakav pristup sistemu, premda se dosta razmišljalo o tome da im se dopusti čitanje poruka itd. (docnije smo uveli demo mod u kome svakome pružamo priliku da, tokom kratke seanse, upozna neke od mogućnosti *Sezama*). Ubrzo se pokazalo da naša nesumnjiva nespokojnost pred „dan D" nije bila potrebna: veliki broj korisnika je odlučio da se pretplati, pa je *Sezam* posle 15. oktobra bio tek nešto manje zauzet nego prethodnih dana, iako je utrostručio broj linija. Činjenica je, naravno, da se broj korisnika sistema praktično prepolovio, ali većina onih koji se nisu pretplatili su bili „povremeni" korisnici koji su pozvali par puta, možda sa pozajmljenog modema, kao i oni „lažni" korisnici koji su služili da se obezbedi dodatno vreme na sistemu. Broj pretplatnika je barem duplo prevazišao naše optimističke prognoze, što je značilo da *Sezam* može nesmetano da raste i širi asortiman svojih usluga.

Jedna od anegdota vezanih za višelinijski rad svakako je promena telefonskog broja – pošto smo želeli da *Sezam* ima samo jedan broj i da se pozivi automatski preusmeravaju na prvu slobodnu liniju, bilo je neophodno da se telefonski broj koji smo do tada koristili promeni. Ovakav zahtev je očito bio pomalo nova stvar za poštu (koja je do sada mnogo doprinela razvoju *Sezama* odobravajući linije kada god su bile potrebne, sarađujući na raznim varijantama povezivanja koje smo izmišljali i, naravno, obezbeđujući pouzdan rad čitavog sistema), pa je broj promenjen „iznanada" – u

Slika 1 Broj poziva



Koliko telefoni zvone: *Sezam* svakog meseca opsluži između 12 i 14000 poziva

Slika 2 Vreme na vezi



Ujednačeno opterećenje: Svakog meseca korisnici na *Sezamu* provedu između 2500 i 3000 časova odnosno 300-400 radnih dana

Slika 3 Vreme u chatu



Najviše se pričalo u aprilu i maju: korisnici *Sezama* provedu u chat-u 75-140 radnih dana

jednom trenutku *Sezam* sa dobijao na broj 647-225 a već u sledećem na 648-955. Za neše konsnike to je bio svojevrstan šok ali se pravi broj brzo „rasčuo" pa je problem zaboravljen. Desilo se međutim da naš stari broj iz tehničkih razloga bude dodaljen nekom privatnom stanu (obično se broj dodeljuje novom pretplatniku tek posle šest meseci) – kako je on (ili ona) prvih dana podneo po više stotina poziva (raspoređanih naročito u noćnim satima) i pištanje sa druge strane linije, možete samo da zamislite.

ŠAHOVSKA OLIMPIJADA

Rastu *Sezama* neobično je doprinela Šahovska olimpijada u Novom Sadu čiji je *Sezam* bio zvanični informacioni servis. To za *Sezam* nije predstavljalo samo priliku za afirmaciju i nabavku opreme – iz čitavog projekta smo izvukli i brojna tehnička saznanja do kojih bi inače bilo teško ili nemoguće doći. Najteži deo posla obavili smo 7. novembra kada smo server naše male mreže (Mrazov AT kompatibilac, 12MHz, 1Mb RAM, 68Mb disk), koji nas je verno služio od *Sezamovih* početaka, poslali u zasluženu „penziju" („penzija" je kratko trajala – ovaj računar, sa malo više memorije i dodatnim diskom, i danas neprekidno radi opslužujući mrežu Novinskog sektora BIGZ-a). Zamenio ga je najpre privremeni 33MHz 386/64K Cache sa 8M RAM-a i IDE diskom od 120 megabajta a onda i Compaq DeskPro 386 na 25 MHz sa diskom od 330 megabajta, server koji je tokom čitave ove godine radio izuzetno pouzdano zadovoljavajući sve potrebe *Sezamove* ekspanzije – nije ipak daleko dan kada će mu trebati još memorije i dodatni disk. Ne mnogo manja promena odigrala se i na području operativnog sistema – mrežu dLINK, koja se dobro pokazala u periodu testiranja višelinijskog *Sezama*, zamenili smo *Novell*-om 2.15 čije su performanse daleko primerenije planiranom redu na 15 nodova. *Sezam* i danas radi na ovoj verziji *Novell*-a premda planiramo da, pre svega zbog dobitka u performansama, uskoro pređemo na verziju 3. DOS 3.30 pod kojim smo do nedavno radili već je zamenjen DOS-om 5.0, pre svega zbog standardizacije operativnog sistema na svim PC kompjuterima koji se koriste u Redakciji „Računara".

Petnaest nodova je, jasno, značilo 15 računara (koristili smo 5 AT računara i 10 286 XT računara sa 8-bitnim mrežnim karticama) čije je fizičko smeštanje u naše prostorije zahtevalo dosta „zemljanih radova", pored ostalog i dovođenje posebnog voda za napajanje lokalne mreže. Čitava instalacija je, pomalo i na naše iznenađenje, potrajala svega 6-7 sati, pre svega zato što je oprema koju smo dobili bila znalački kompletirana i pripremljena, ako je bilo problema, bilo ih je jedino sa modemima.

Promena hardvera i operativnog sistema, na sreću, nije zahtevala veće izmene u softveru koji čini *Sezam* – centralni program PBB je pretrpeo nekoliko manjih izmena i prilagođenja i razvijena su eksterni programi CHESS (održavanje baze šahovskih partija) i POBOX (slanje privatne pošte zvaničnim licima i učesnicima Olimpijade). Programiranja, dakle, nije bilo previše ali je priprema konfiguracionih fajlova zahtevala mnogo vremena i još više pažnje – trebalo je napraviti kompletno novi sistem menija za korisnike koji nas zovu preko „šahovske" grane sistema a onda te menije uklopili u standardne menije naših registrovanih korisnika. Organizacija Olimpijade nas je, osim toga, konačno „naterala" da *Sezamov* korisnički interfejs prevedemo i na engleski jezik (sa *Sezamom* se sada može „razgovarati" na srpskohrvatskom, slovenačkom, makedonskom ili na engleskom), što je davnašnji plan više puta odlagan zbog nedostatka vremena. Trebalo je, najzad, obezbediti da novim korisnicima koji zovu „šahovski" broj 648-899 (na koma je bilo 10 linija) pristup bude omogućen odmah (a ne posle plaćanja pretplate), ali i ograničen na osnovne operacije kao što je privatna pošta i sakupljanje novih partija odnosno rezultata, dok bi „pridošice" na naš standardni broj 648-955 prolazile kroz uobičajeni postupak registracije.

Prvo što smo zapazili kada smo instalirali sistem bilo je **strahovito** poboljšanje performansi – iako je radio na 5 puta više linija, tekst i poruke su prosto letele po ekranu a čak su i potencijalno dugotrajna pakovanja poruka postala operacija jedva vredna pomena. Došli smo do zaključka da zasluge za ovo poboljšanje pre svega ima višestruko povećan RAM u serveru i *Novell* umesto *dLINK*-a, a tek onda 386 umesto 286 mašine. Proračuni govore da bi ovakav server mogao opslužiti i 32 noda.

Zanimljivo je bilo raditi i sa dve „zakupljene linije" preko kojih smo 24 časa dnevno bili povezani sa Novim Sadom – specijalitet ovih linija je činjenica da ne treba zvati pošto sa veza

TANJUG SEZAM BBS

SEZAMOVO „DRUGO LICE"

Dok prebire po konferencijama i direktorijumima Sezama, ostavlja ili preuzima elektronsku poštu, ili „čavrlja" u chat-u, najveći broj pretplatnika i ne razmišlja o tome – a mnogi su propustili čak i da to saznaju – da se „s one strane" te osnovne aktivnosti modemskog sistema „Računara" odvija jedan strogo definisan, profesionalni informatički posao visokog nivoa.

Naravno, „s one strane" je samo metafora, pošto se radi o istovremenom poslu, „isprepletenom" sa onim osnovnim. Reč je o sistemu TANJUG SEZAM BBS, koji u okviru istog „sezamskog" hardvera i softvera opslužuje svoje vlastite korisnike. Taj paralelni posao nije namenjen pretplatnicima Sezama (mada i oni mogu da koriste servis vesti ako to regulišu sa Tanjugom) nego redakcijama, slobodnim novinarima, ambasadama, stranim dopisništvima, klubovima naših iseljenika u inostranstvu i svima onima koji po prirodi svoga posla ili interesovanja prate tekuće agencijske vesti. Korisnici Tanjug Sezama plaćaju agenciji pristup vestima i njihovo korišćenje u srazmeri sa količinom preuzetog materijala u određenom vremenskom periodu. Pristup tom informacionom sistemu, korisnički status, finansijske odnose i sve drugo regulišu direktno sa Tanjugom (Marketing: 011/625-722, 332-231), a uprava Sezama zadužena je za stručno-tehnički razvoj sistema i njegovo održavanje.

Sa realizacijom ovog značajnog računarsko-modemskog informatičkog projekta krenulo se početkom ove godine. Prema zajednički utvrđenoj koncepciji i metodologiji rada sistema, razvijen je odgovarajući softver i nakon tri meseca probnog rada sistem je krajem maja, kako bi se reklo birokratskim jezikom, „ušao u redovnu eksploataciju".

Posredstvom posebnog kanala (takozvane „poprečne" vodovne veze) iz Tanjuga u Sezam neprekidno pristižu vesti grupisane u sedam takozvanih „servisa": generalni servis Tanjuga na srpskohrvatskom jeziku (zbog maksimalno široke tematike i namene ovaj servis se najviše koristi), svetski servis na engleskom, servisi na španskom, francuskom i nemačkom jeziku, dnevni teleks servis i servis ekonomskih informacija. Iskustvo pokazuje da je za takve vesti optimalno vreme korišćenja sedam dana, posle čega njihova upotrebna vrednost postaje bitno umanjena. Da ne bi nepotrebno opterećivale sistem, Sezam sve vesti nakon sedam dana „života" automatski briše.

U odnosu na druge „kanale" kojima se Tanjugove vesti plasiraju u zemlji i svetu Tanjug Sezam pruža dve bitne prednosti.

Prva je oblik u kome vesti stižu „na sto" korisnika: one su na hard disku (ili disketi) na raspolaganju za eventualno preoblikovanje, brzo štampanje (ako se čitaju u informativnim emisijama ili šalju na slaganje), direktno prelamanje teksta (ako štamparija raspolaže kompjuterskim slogom), delimično preuzimanje radi kompiliranja tekstova, citiranje kod pisanja na računaru.

Druga prednost je izvanredna mogućnost pretraživanja vesti. Standardni metod pregleda podrazumeva da se nakon ulaska u konferenciju NEWS bira servis, a zatim vremenski raspon iz koga se vesti pozivaju radi pregleda. Tokom pregleda, vesti se po želji presnimavaju. Moguće je, naravno, i preuzimanje vesti bez detaljnijeg pregleda. Manje standardan, ali „zanimljiviji" i krajnje efikasan metod je pretraživanje po želji. U zadatom vremenskom rasponu vesti mogu da se selekcionišu i izdvoje po bilo kom ključu prepoznavanja: imenu ličnosti (na primer: Karrington), ili grada (Hag), ili bilo kom drugom pojmu (tord, granata) ili delu pojma. Naravno, kod traženja je cilj da se navede pojam koji ne predstavlja koren ili deo nekih drugih reči. Tražena reč „rat", na primer, izdvojiće i vesti u kojima postoji neka od reči „karata", „prevrat", „vratoloman" i tome slično. Ako se želi izdvajanje jedne određene specifične vesti, potrebno je navesti kakvu precizniju formulaciju za koju znamo da se u njoj nalazi („lord Karrington je jucce"). Razumljivo, mora se voditi računa o uobičajenom načinu pisanja nekih domaćih slova („š" se piše kao „ss", „č" kao „cc"...) i fonetskom pisanju stranih imena (Tacher, Genshar...).

Tanjug Sezam je danas u punom pogonu i obavlja jednu neosporno važnu informatičku ulogu. Nekim korisnicima koji imaju imaju uhodanu direktnu (kablovsku, teleks ili telefaks) vezu sa Tanjugom, kao što su televizijske i novinske kuće, Sezam predstavlja dopunski ili rezervni sistem za slučaj da im otkaže osnovni. Ali u poslednje vreme sve je više korisnika kojima je Tanjug Sezam jedini kanal do Tanjugovih servisa.

Jedini – ali vredan.

Esad Jakupović

Statistika Sezama - 11.11.89 – 11.11.91

Konferencija	Poruka	Dužina	Datoteka	Dužina
UPUTSTVO	155	274877	0	0
NOVOSTI	1636	5284675	7	179733
SAJAM	47	45788	0	0
RACUNARI	947	985443	5	93255
ŠAH	3533	3113637	27	485555
KOMUNIKACIJE	3440	3280838	117	3378483
PC HARD	2788	2393678	45	1414588
PC SOFT	4757	4287745	313	8881881
PC OS	239	190107	8	47341
PC PROG	498	448393	24	555089
PC USER	952	428348	22	825570
ATARI ST	4138	11359187	475	7715837
AMIGA	477	338800	28	774189
OKNA	2878	2885805	91	3582858
FORUM	17113	20837128	51	1818385
EXTRA	1587	1478887	36	1574882
VICEVI	3144	3445577	29	371394
MALI OGLASI	1730	.855571	10	145825
HOST	251	207898	9	833315
Ukupno:	49897	61584631	1348	38557473
Privatne poruke:	82884	194847488		
grupe:	5821	4583118		

Broj poziva	182801
On-line vreme	39372:58:13
Chat vreme	7329:15:42*
Upload datoteka	21521*
Upload kb	231836
Download datoteka	119810*
Download kb	3817487

* Podatak se odnosi na razdoblje 01.03.91–11.11.91

Najviše poziva

Br.	Pseudonim	Prezime i ime	Grad	Poziva
1	vcalic	Vlada Čalić	BEOGRAD	2464
2	bulaja	Jovan Bulajić	BEOGRAD	2432
3	belinda	Dejan Belinda	BEOGRAD	2312
4	ppekovic	Pavle Peković	SREMČICA	2074
5	princess	Aleksandra Kostić	BEOGRAD	1785
6	jvalant	Janez Valant	BLED	1757
7	yupc	Goran Vuinsić	KRAGUJEVAC	1558
8	miha	Siniša Mihailović	NOVI BEOGRAD	1557
9	ddimcic	Dejan Dimčić	BEOGRAD	1530
10	bois	Boško Radivojević	NOVI BEOGRAD	1507

Najduže na vezi

RB	Pseudonim	Prezime i ime	Grad	HHH:MM:SS
1.	vcalic	Vlada Čalić	BEOGRAD	584:35:51
2.	vesna	Vesna Tulović	BEOGRAD	558:14:51
3.	bulaja	Jovan Bulajić	BEOGRAD	478:58:17
4.	dvesalinovic	Dejan Veselinović	BEOGRAD	458:18:58
5.	ppekovic	Pavle Peković	SREMČICA	429:37:50
6	alsma	Aleksandar Simonjanović	BEOGRAD	428:35:44
7	princess	Aleksandra Kostić	BEOGRAD	419:58:40
8.	kale	Slobodan Kalešić	BEOGRAD	393:39:44
9	tamara	Tamara Mihailović	NOVI BEOGRAD	353:57:54
10	boja	Bojan Stanojević	NOVI BEOGRAD	353:02:14
11	lanik	Dalibor Lanik	BEOGRAD	389:45:17
12	mjova	Miljan Jovanović	BEOGRAD	333:43:07
13	jvalant	Janez Valant	BLED	323:59:55
14	belinda	Dejan Belinda	BEOGRAD	322:25:18
15	ddimcic	Dejan Dimčić	BEOGRAD	321:58:35

Najduže u chat-u

RB	Pseudonim	Prezime i ime	Grad	HHH:MM:SS
1	vesna	Vesna Tulović	BEOGRAD	379:38:54
2	princess	Aleksandra Kostić	BEOGRAD	218:05:25
3	vcalic	Vlada Čalić	BEOGRAD	179:18:51
4	sjankovic	Srđan Janković	BEOGRAD	178:54:21
5	alexa	Aleksandar Stanjanović	BEOGRAD	158:25:35
6	prvul	Miloš Prvulović	BEOGRAD	153:15:27
7	tamara	Tamara Mihailović	NOVI BEOGRAD	138:27:58
8	bois	Boško Radivojević	NOVI BEOGRAD	136:25:48
9	dadad	Dačad Dimitrijević	BEOGRAD	131:11:03
10	zdimic	Zoran Dimić	BEOGRAD	118:53:13

Najviše poruka u konferencijama

RB	Pseudonim	Prezime i ime	Grad	Poruka
1	madamov	Milan Adamov	BEOGRAD	2157
2	belinda	Dejan Belinda	BEOGRAD	1471
3	dvesalinovic	Dejan Veselinović	BEOGRAD	1533
4	lanik	Dalibor Lanik	BEOGRAD	1454
5	bulaja	Jovan Bulajić	BEOGRAD	1301
6	vcalic	Vlada Čalić	BEOGRAD	1012
7	ppekovic	Pavle Peković	SREMČICA	1009
8	kale	Slobodan Kalešić	BEOGRAD	938
9	mkiric	Marko Kirić	BEOGRAD	901
10	vkostic	Vlada Kostić	BEOGRAD	783
11	dpozaric	Dražen Požarić	POREČ	875

Najviše poslatih datoteka

Br.	Pseudonim	Prezime i ime	Grad	Kilobajta
1	madamov	Milan Adamov	BEOGRAD	22818
2	bulaja	Jovan Bulajić	BEOGRAD	18855
3	lanik	Dalibor Lanik	BEOGRAD	18815
4	ppekovic	Pavle Peković	SREMČICA	9898
5	nboskovic	Nikola Bošković	BEOGRAD	7438
6	drabes	Dragan Petrović	BEOGRAD	5147
7	dvesalinovic	Dejan Veselinović	BEOGRAD	3878
8	bmnax	Branislav Mijatović	BEOGRAD	3889
9	alsma	Aleksandar Simonjanović	BEOGRAD	3288
10	mjova	Miljan Jovanović	BEOGRAD	3283
11	bale	Aleksandar Bakić	BEOGRAD	3258
12	skuzovac	Srđan Kuzovac	BEOGRAD	2753
13	dsala	Dragan Simić	SREM. KAMENICA	2785
14	belinda	Dejan Belinda	BEOGRAD	2527
15	jmedli	Jilaly Medli	NOVI BEOGRAD	2485
16	zormi	Zoran Milosavljević	ZEMUN	2413
17	neša	Nenad Veljković	BEOGRAD	2253
18	apko	Aleksandra Petrović	BEOGRAD	1935

Najviše primljenih datoteka

RB	Pseudonim	Prezime i ime	Grad	Kilobajta
1	bora	Bora Živanović	BEOGRAD	47555
2	fsnike	Slobodan Petrović	NOVI BEOGRAD	48225
3	vcalic	Vlada Čalić	BEOGRAD	37177
4	nvidovic	Nenad Vidović	BEOGRAD	38518
5	lanik	Dalibor Lanik	BEOGRAD	35557
6	deakra	Dragan Petrović	BEOGRAD	32888
7	dangsanj	Dragan Jeremić	ZEMUN	31400
8	mnikolic	Miloš Nikolić	BEOGRAD	29800
9	madamov	Milan Adamov	BEOGRAD	28158
10	rkorda	Radovan Korda	BEOGRAD	27581
11	zddb	Zoran Krkić	BEOGRAD	30087
12	braco	Jasmin Šipanov	BEOGRAD	28185
13	bulaja	Jovan Bulajić	BEOGRAD	25338
14	mkrkic	Miloš Krkić	BEOGRAD	25553
15	nesla	Srboljub Našić	NIŠ	25212

ni ne prekida! Obzirom da nam je bio potreban brz i pouzdan prenos podataka, na ove linije smo stavili kvalitetne *Compaq*-ove računare.

Čitav tok Olimpijade oprema je herojski podnela – osim jednog večernjeg nestanka struje u čitavom BIGZ-u prvog dana rada. *Sezam* je tokom čitavog meseca radio 24 sata dnevno bez minuta prekida, čak smo i *backup* (koji obično zahteva gašenje sistema) vršili „u hodu" uz pomoć specijalno pripremljenog softvera i zahvaljujući obilju prostora na disku (nešto kasnije nabavili smo i 250M strimer koji je čitavu operaciju dalje ubrzao). Posle nekoliko dana „uhodavanja" partije i turnirske tabele su stizale svega nekoliko časova po završetku kola tako da je *Sezam* bio ono što smo i želeli da bude – najbrži način da se dođe do informacija sa šahovske Olimpijade.

Najviše su uživali naši registrovani korisnici – povećali smo dnevno vreme na 200 odnosno 120 minuta i svi zajedno otkrili draži *chat*-a na 13 nodova. Nije bilo popodneva, večera i noći bez modemske „žurke" tokom koje bi se diskutovalo o raznim računarskim, šahovskim, neračunarskim i nešahovskim temama. Kada su novosadske hostese sa Info pultova naučile šta je *chat* bilo je urnebesa, a priča se i o nekim ekskurzijama do Novog Sada radi ličnog upoznavanja. Najzad, pozovite *Sezam* i pročitajte sve tračeve o tome!

Izraženo brojkama, period od 7. novembra do 8. decembra nam je doneo preko 2000 poruka u konferencijama (4 megabajta) i 2500 privatnih poruka – pokazalo se zapravo da dodavanje svake nove linije ne obogaćuje sistem samo kvantitativno nego i kvalitativno, a jasno su uočljive i velike razlike između klasičnog BBS-a i profesionalnog kompjuterskog informacionog sistema.

ZAŠTO KASNI UPUTSTVO

A problemi, zapitaće svaki čitalac ovih redova koji nije zvao *Sezam*: ovoliki projekat nije mogao proći bez problema? Začudo (ili možda ne začudo, ako se uzme u obzir uloženi trud) ozbiljnih problema gotovo da nije bilo. Prvih dana nas je mučio jedan od nodova koji je proizvodio ogromne količine „đubreta" i česte prekide veze odnosno „upadanja" u vezu drugih korisnika (na *Sezamu* je tim povodom no-

Slika 4 Konferencijskih poruka



Sve više i više broj poruka koje pristižu u *Sezam*ove konferencije raste iz meseca u mesec

Slika 5 Poruka u FORUM-u



Živimo u uzbudljivo vreme prema broju poruka u konferenciji FORUM lako se vidi kada su dramatični događaji u zemlji dostigli vrhunac.

Slika 6 Upload (megabajta)



(Ne)stabilan uspon I pored umerenih varijacija, iz meseca u mesec *Sezam* od korisnika prima sve više datoteka

vokomponovano i nekoliko poslovica) ali je sve nestalo kao rukom odneseno kada je u pošli „nešto prevezeno drugačije". Par puta se osim toga dešavalo da se rad nekog noda **strahovito** uspori (utisak otprilike kao kada predete sa PC-ja na ZX81) što se uglavnom lečilo njegovim resetovanjem, pomoglo je i kada smo jedan od terminatora na mreži uzemljili.

Sa tehničkim problemima se, uostalom, moglo oditi ali su nas namučili neki nesavesni korisnici koji su se radi dobijanja dodatnog vremena za *chat* prijavljivali pod izmišljenim imenima (često i stranim) zbog čega nam je praktično nemoguće da procenimo koliko je za vreme trajanja Olimpijade stvarno bilo poziva iz zemlje a koliko iz sveta. Mučili su nas, osim toga, i stalni pokušaji „pogađanja" lozinki – nepoznati „hakeri" su izgleda odustali od lozinki službenih lica *Sezama* (pošto im je valjda očigledno da ih ne mogu pogoditi) ali su se zato orijentisali na registrovane korisnike sa ciljem da im kradu vreme i pristupaju sistemu bez plaćanja pretplate. Pokazalo se da oni koji „gađaju" lozinke to ne rade nasumice već da za svakoga smišljeno isprobavaju njegovu lozinku sa drugih BBS-ova što, pre svega, znači da te lozinke „cure" nekim kanalima (protiv čega mi, na žalost, ne možemo ništa preduzeti) ali i to da korisnici u sopstvenom interesu treba da biraju netrivijalne lozinke i da ih ne koriste na drugim BBS-ovima.

Najveći „problem" je, međutim, činjenica da sve što je lepo ima kraj – kada se Olimpijada završila, uvideli smo da bi povratak na tri linije bio previše surova stvarnost. Zato je posle desetak dana *Sezam* dobio šest linija što je, prema našim proračunima, predstavljalo optimalnu ravnotežu između opterećenosti sistema i ekonomskih mogućnosti. Taj broj linija je bio dovoljan negde do maja 1991, kada je vreme potrebno za dobijanje *Sezama* počelo drastično da raste, pre svega pod pritiskom novih korisnika. S pravom smatrajući da će opterećenje tokom letnjih meseci biti nešto manje, pripremali smo se za proširenje kapaciteta od jeseni. I zaista, počevši od 25. oktobra 1991. *Sezam* radi na 10 linija vezanih na broj 648 899. 10 linija je za sada (a nadamo se i za bližu budućnost) dovoljno, a kada pritisak tokom sledeće godine ponovo poraste... mislićemo i o proširenjima. 150 linija američkog EXEC PC, najvećeg BBS-a na svetu, su i dalje daleko, ali im se malo po malo i približavamo, čak i u svetskim merilima, sistem za modemske komunikacije koji radi na 10 linija je sasvim pristojna referenca!

Šahovska olimpijada je zahtevala prilično angažovanja tako da je jedan od dugova korisnicima pomalo odložen – Uputstvo za upotrebu *Sezama* u vidu atraktivno štampane knjiži-

Slika 7 Download (megabajta)



Veliko interesovanje. Iako je *Sezam* bazično konferencijski sistem, sa njega se preuzima sve više i više datoteka. Stalno rastući, download se približava granici od 500 megabajta mesečno.

ce na 200 strana, planirano je za 15. oktobar a štampano tek tokom decembra tako da su ga pretplatnici dobili kao mali novogodišnji poklon. Uputstvo je omogućilo apsolutnim početnicima da uspešno ostvare vezu sa *Sezamom* a takođe i iskusnijim korisnicima da kreativno primene sve komande koje smo uveli i tako uz minimalne telefonske račune prenesu ono što ih zanima. Iako smo u startu štampali Uputstvo u prilično velikom tiražu, prvo izdanje je u dane kada ovo pišemo poprilično iscrpljeno pa polako razmišljamo o drugom – treba, najzad, dokumentovati i razne dodatke *Sezamovom* korisničkom interfejsu, koji su uvedeni tokom 1991. godine!

REVIZIJA SOFTVERA

Glavni novitet bilo je uvođenje grupe. Od prvih dana rada *Sezam*, za razliku od većine drugih BBS-ova, omogućava grupi korisnika koji sarađuju po nekom osnovu da se dogovore o šifri i da onda šalju poruke na nju – ko god zna šifru može da čita poruke! Rast *Sezama* je pokazao da korisnike ovakva „šifrovana pošta" veoma zanima, ali da usvojeni koncept ima i ozbiljne slabosti – privatna pošta je prekomerno rasla što je usporavalo rad a šifre su postale precest predmet „pogađanja" i hakerisanja. Zato smo krajem aprila stvari iz temelja promenili – šifre u nekadašnjem obliku su ukinute ali je zato omogućeno svakom korisniku da otvori tri grupe (privatne konferencije) i u njih „pusti" po najviše 12 članova. Svaki član neke od grupa bez problema čita i šalje poruke u nju dok je onima koji nisu članovi pristup nemoguć, čak i ako znaju naziv (šifru) grupe.

Novina je veoma dobro primljena uz „očekivanu" primedbu da je 12 korisnika malo i stalne zahteve da se za pojedine „važne" grupe odobri i više (npr. za grupu *devojke* pošta na *Sezamu* odavno ima više od 12 pripadnica lepšeg pola). Na ovakvo povećanje nismo mogli da se odlučimo kako iz tehničkih tako i iz organizacionih razloga. Tehničkih, zato što bi povećanje na, recimo, 20 članova znatno usporilo naredbu *group list* koja se, pored ostalog, izvršava kada se korisnik prijavi na sistem. Organizacionih, zato što nam je želja da grupe ostanu to što jesu: način da nekoliko ljudi prodiskutuje neku stvar od zajedničkog interesa, a nikako da postanu zamena za konferencije. Kada bi grupe mogle neograničeno da rastu, pojavila bi se tendencija da se mnoge „osetljivije" diskusije povlače iz konferencija što bi bilo prilično neprijatno, posebno sa aspekta novog korisnika – uz ogroman saobraćaj poruka na *Sezamu* on ne bi imao šta da čita jer ga niko ne bi poznavao pa ne bi bio učlanjen ni u jednu grupu! Posle početnog oduševljenja grupama i osnivanja „sekcija" za sve i svašta, stvari su docnije došle u „ravnotežno stanje" pa se grupe koriste za ono za šta su i namenjene. Vredi dodati i da su neke grupe za koje je bilo dosta interesovanja (npr. *sf* za naučnu fantastiku, *flight sim* za čuveni program za simulaciju leta itd) vremenom prerasle u teme u odgovarajućim konferencijama.

Ostale izmene u korisničkom interfejsu bile su manjeg obima – uveli smo mogućnost da više korisnika vodi privatni razgovor skriven od očiju ostalih prisutnih u *chat*-u (u početku odušavljeno prihvaćen, ovaj modalitet se danas relativno retko koristi ali će „doći na svoje" kada broj linija toliko poraste da *chat* svih na jednom kanalu postane pretežak za praćenje – i sada se to povremeno dešava na 10 linija), zatim opciju kojom se može videti kada je neka datoteka poslana na *Sezam* i koliko je puta *download*-ovana, brzi pregled onih konferencija u kojima ima novih poruka i par drugih sitnica. Od maja 1991. osnovni program nije menjan – radili smo samo na usavršavanju i dodavanju pojedinih eksternih programa. To, naravno, ne znači da smo stigli do kraja – uskoro se planira još jedna revizija korisničkog interfejsa u okviru koje će, na primer, biti omogućeno da se datoteke iz PAD-a direktno prosleđuju drugim korisnicima ili povezuju sa porukama bez potrebe da se radi novi *upload*, zatim uvedeni prošireni opisi datoteka u direktorijumima uz pomoć posebne konferencije i tako dalje – sva ova unapređenja zasnovana su na predlozima korisnika i proučavanju sistemskih dnevnika. Pa ipak, u osnovi se radi o „finom štelovanju" softvera – koncepcija *Sezama* je sa tehničke strane dovedena čak i dalje od tačke koju smo, kada smo počeli da radimo na projektu, mogli da zamislimo. Dalje izmene će se verovatno svoditi uglavnom na uvođenje novih usluga.

UMREŽENI SEZAM

Osnovni novitet na kome smo tokom 1991 godine radili je umrežavanje *Sezama*. Čitav svet komunikacija uostalom leži tome – ideja je da pozovete jedan sistem preko koga ćete po potrebi doći u vezu sa nekim drugim sistemom i iskoristiti njegove usluge. Preko tog drugog sistema možete doći do nekog trećeg i tako dalje – ukratko, vaš lokalni telefonski poziv omogućava vam kontakt sa čitavim svetom koji postaje, da se poslužimo tim već previše rabljenim izrazom, „globalno selo".

Problem umrežavanja „napali" smo sa dve strane. Prva od njih, nazvana *SezamNET*, zahtevala je mnogo razvoja – zamislili smo mrežu ekspozitura *Sezama* u raznim krajevima zemlje. Svaka od tih ekspozitura, prema potrebi, može da bude mali BBS ili sistem sa više telefonskih linija, i svi oni za onoga ko pozove izgledaju jednako – iste konferencije, iste poruke, iste datoteke... Kako bi se ovo stanje azuriralo, povremeno se vrši razmena pošte između čvorova *SezamNET*-a. Ova razmena može da teče neprekidno ukoliko je neki od čvorova dovoljno veliki i značajan da opravda investiciju u 24-časovnu direktnu telefonsku vezu, za manje čvorove razmena bi se odvijala par puta dnevno tako što jedan čvor pozove centralni (ili neki drugi, već prema pulanji koja najmanje košta) i prenese mu sve u međuvremenu pristigle konferencijske i privatne poruke kao i datoteke, primajući istovremeno ono što su dobili drugi čvorovi (protokol kao stvoren za takve primene zove se BiModem – njega smo i koristili).

Već na prvi pogled vide se neki od problema koje treba rešiti da bi ovakva mreža korektno funkcionisala. Na jednom čvoru poruka u konferenciji nosi, na primer, broj 4.125 a na drugom, gde je stigla nekoliko sati kasnije (pri razmeni pošte), 4.132. Kada na drugom čvoru korisnik zada REPLY 4.132, u zaglavlju poruke koja će nastati treba da piše „odgovor na 4.132". Kada ta poruka bude poslata na prvi čvor, to zaglavlje treba da glasi „odgovor na 4.125" dok će na nekom trećem nodu to možda biti „odgovor na 4.120". Problema tog tipa ima jako mnogo – većinu smo uspeli da percepiramo dok smo postavljali osnove novog softvera, neki su iskrsli u toku samog kodiranja a poneki je „promakao" i za fazu testiranja. Da bismo to testiranje što detaljnije obavili, bio nam je potreban barem još jedan čvor *Sezama*. U pomoć nam je priteknao Saša Tapavički koji je gotovo pola godine održavao eksperimentalni nod u Novom Sadu trudeći se da izleči ili barem prijavi sve „dečije bolesti" našeg novog sistema. Mnogi korisnici iz Vojvodine su se opredelili za zvanje ovog čvora umanjujući tako svoje telefonske račune. Beograd je u takvoj situaciji imalo smisla zvati samo povremeno, zbog *chat*-a.

Razmena poruka obavljana je 2 puta dnevno, pri čemu se kao glavni problem pokazao trenutak dizanja slušalice. Kada je *Sezam* ozbiljno opterećen, ni jednom od njegovih nodova nije lako da pozove neki telefonski broj; nod, na primer, utvrdi da trenutno telefon ne zvoni, podigne slušalicu ali baš u tom trenutku stigne poziv! Neko će reći da je verovatnoća takvog događaja jako mala (tako smo i mi mislili dok nismo probali) ali uopšte nije tako – dobar deo naših prenosa u početku nije uspevao baš zbog ovog neverovatnog slučaja. Pošto smo se uverili da američke centrale nisu džabe izmislile takozvani *dial tone* koji modemi prepoznaju i zazalili što naše centrale emituju sasvim drugačiji signal, ostalo nam je samo da sa služimo trikovima. Složenim testiranjem RING linije (koju takođe nemaju svi modemi) uspeli smo da sa priličnom sigurnošću lociramo trenutak kada je linija slobodna što smo i testirali smestivši razmenu pošte u najopterećenije vreme, dakle u 12 noću. Radilo je.

Pošto je razvoj *SezamNET* softvera uspešno priveden kraju, došlo je vreme da se razmisli o organizacionim pitanjima pa smo u majskim „Računarima" objavili poziv zainteresovanima za osnivanje *Sezam* čvorova. Iako je taj poziv bio više „pilot", interesovanje je prevazišlo sva očekivanja – izgledalo nam je da ćemo vrlo brzo imati ekspoziture u svim krajevima zemlje. Na žalost, najpre dramatičan a zatim tragičan razvoj događaja poslednjih meseci morao je odbiti čitav posao – kako pre svega da osnivamo čvorove u raznim krajevima zemlje kada se ne zna dokle se ta zemlja prostire? Kako da razmenjujemo poruke kada telefonske veze svaki čas prestaju da funkcionišu? I kako da u ovako dramatičnom i nesigurnom trenutku uđemo u ovu za *Sezam* veliku investiciju? Zato je čitav *SezamNET* projekat, tehnički potpuno realizovan, morao da bude odložen za neko malo srećnije vreme koje, iskreno se nadamo, neće morati dugo da se čeka.

Drugi pristup umrežavanju nije ništa manje ambiciozan – želeli smo da *Sezam* povežemo sa nekom globalnom kompjuterskom mrežom, pri čemu smo od početka želeli da to bude neka profesionalna organizacija a ne mreža amaterskih BBS-ova kao što je na primer FidoNET. Nije trebalo daleko tražiti – mreža koja će omogućiti našim korisnicima da šalju poštu po čitavom svetu, prate brojne strane konferencije i preuzimaju interesantne datoteke sa brojnih *trickle* servera je Bitnet koji se u Evropi zove EARN. Da bi stvari bile još lepše, vremenom ćemo omogućiti korisnicima da preko Bitnet-a šalju *Sezamu* svoje EXEC-e a zatim istim putem primaju PAD-ove što bi, na primer, Jugoslovenima u akademskim sredinama USA omogućilo da komuniciraju sa *Sezamom* praktično bez ikakvih telefonskih troškova.

Povezivanje *Sezama* sa EARN-om je u poodmakloj fazi, kako organizaciono tako i tehnički. Nije nam običaj da u ovakvim najavama preciziramo neke rokove, pošto uvek ima stvari koje su izvan naše kontrole, ali verujemo da nije daleko dan kada će pošta sa *Sezama* elektronskom brzinom putovati po svetu. Naravno, i priključak na paketnu mrežu (JUPAK) je nešto o čemu se razmišlja, ali je zbog visokih cena (kako samog priključka tako i potrebnog hardvera) to nekakav srednjoročni plan, možda za kraj 1992. godine. Ukratko rečeno, tokom čitave 1991. radili smo na raznim vidovima umrežavanja *Sezama* ali će većina toga doći do izražaja tek tokom sledeće godine – nadamo se da će, kao što je 1991. prošla u znaku rasta broja korisnika, poruka i linija, 1992. biti u znaku povezivanja *Sezama* na raznim kompjuterskim mrežama.

SVAKODNEVNI SEZAM

Najznačajnija nova usluga koju je *Sezam* ponudio su Tanjugovi servisi, o čemu objavljujemo poseban prilog u okviru ovog teksta. Još jedan značajan projekat koji smo ove godine preduzeli je SOR, *off line reader* za *Sezam*. SOR je napisao Zoran Živolić, a tekuća verzija se, potpuno besplatno, može preuzeti iz COM direktorijuma *Sezama*. Uputstvo za upotrebu SOR-a možete naći već u ovom broju *Računara*.

Sve ove akcije, naravno, neće baciti u zasenak ono najlepše na *Sezamu* a to je svakodnevna živost u konferencijama, u grupama, u direktorijumima, u *chat*-u... Tokom 1991. pokrenute su neke nove konferencije i uvedene brojne nove teme u postojećim. Trenutno su javnosti pristupačne kako konferencije „opšteg" tipa, tako i one specijalizovane. U konferencije opšteg tipa ubrajamo UPUTSTVO (kompletan sadržaj Uputstva za upotrebu *Sezama*, sa svim izmenama i dopunama koje unosimo kada *Sezam* dobije neku novu naredbu ili opciju), NOVOSTI (kompjuterske vesti sa stranih mreža), RAČUNARI (diskusija o našem časopisu), SEZAM (razgovori o sistemu, njegovim mogućnostima, upotrebi, budućnosti...), KOMUNIKACIJE (o modemima, drugim BBS-ovima i mrežama, elektronskoj pošti, radio vezama i sličnim temama), ORKA (opšta računarska konferencija), FORUM (nekada konferencija koja se bavila svim ne-računarskim temama, a od sredine 1991. uglavnom „politička" konferencija, na radost nas baš previše što živimo u vremenima u kojima ova konferencija ima ubedljivo najviše poruka), EXTRA („ostale" teme koje obuhvataju literaturu, sport, film, naučnu fantastiku, vazduhoplovstvo itd), VICEVI (veoma posećena konferencija koju nema potrebe posebno opisivati) i MALI OGLASI.

Značajan deo *Sezama* predstavljaju računarski specifične konferencije. Obzirom da najveći deo korisnika *Sezama* ima PC računare, ovaj segment konferencijskog sistema je posebno razgranat: konferencija PC.HARD bavi se hardverom PC računara, PC.OS operativnim sistemima (DOS, Windows, Unix, OS/2, mrežni softver itd), PC.PROG programiranjem na PC-ju (naročito su atraktivne teme koje se bave C-om, Clipper-om i drugim programskim jezicima) a PC.USER korisničkim aspektima PC računara (tekst procesori, baze podataka, programi za crtanje itd). Specijalizovane konferencije imaju i korisnici Atari ST računara kao i, od jeseni 1991, korisnici Amige.

Unekoliko poseban slučaj su komercijalne konferencije koje mogu iznajmiti preduzeća u cilju reklamiranja svojih proizvoda, ispitivanja tržišta, podrške kupcima i tako dalje. U ovom trenutku jedina komercijalna konferencija pripada firmi MOST – obzirom na veoma povoljne reakcije koje je konferencija izazvala i veliki broj poruka u njoj, verujemo da će se uskoro i drugi proizvođači i prodavci računarske opreme odlučiti na ovakav korak. U svakom slučaju, tokom sledeće godine će se dosta raditi na popularizaciji oglašavanja na *Sezamu* i drugih komercijalnih usluga sistema.

Za veliki broj poruka u konferencijama svakako su veoma zaslužni i moderatori – pošto je već posle nekoliko meseci *Sezamovog* rada postalo jasno da poruka jednostavno ima previše da bi administratori sistema mogli da ih prate, odlučili smo da svaka konferencija dobije svog moderatora, stručnjaka za oblast o kojoj se radi, koji će odgovarati na pitanja, stimulisati i pokretati diskusiju, premeštati „zalutale" poruke iz teme u temu i uopšte pomagati korisnicima da se u datoj konferenciji što odje snađu. Moderatori (čija imena uvek možete pročitati u impresumu „Računara") su se izvanredno snašli u čitavom poslu, tako da u toku 1991. na sistemu praktično nismo ni imali ekscesnih situacija (još se sećamo afere „Ilija Matić" koja je uzdrmala čitav *Sezam* i nekako se privela kraju tek u junu 1990). I pored „varnica" koja su sevale u pojedinim konferencijama (naročito u FORUM-u), diskusija je uglavnom ostajala u domenu korektnog.

Iako *Sezam* od svog početka nije koncipiran kao BBS okrenut *download*-u, 1991. nam je donela veliki broj novih datoteka. Radi se o probranom i dobro testiranom *public domain* i



shareware softveru sa svih strane sveta. Posebnu pažnju posvećujemo programima domaćih autora – tu su kriterijumi, razume se, nešto umereniji, ali nam je godina koja je za nama svejedno donela nekoliko izuzetno zanimljivih programa: pomenimo samo HYUTIL Dragana Zakića (grupa uslužnih programa koji, pored ostalog, omogućavaju vlasnicima Hercules kartice da posmatraju YU slova bez promene EPROM-a), YUFOG G. Biočića (YU slova na VGA karticama po bilo kom rasporedu, uz editor fontova) i VKEYB Vlade Koslića (univerzalni generator drajvera za tastaturu). Od stranih programa najviše pažnje izazvale su nove verzije programa SCAN (detektor virusa) koje su stizale na *Sezam* samo par dana (a često i sati) posle svetske premijere, novi arhiveri (posebno ARJ), komunikacioni paketi (posebno *Telix* i *Telemate*), razni tekstovi o pojedinim slabo dokumentovanim mogućnostima PC-ja (npr. opis svih (?) nedokumentovanih INT poziva) i tome slično.

U cilju lakšeg snalaženja, datoteke su podeljene u veći broj direktorijuma – osnova svega su direktorijumi IBMPC, COM (komunikacioni programi), UNIX, ATARIST, NOVOSTI, RSOFT (programi iz „Računara") i PITALICE, dok je na primer IBMPC direktorijum podeljen na poddirektorijume ARCHIVER, DEMO, DRIVERS (uvek potrebni drajveri za grafičke kartice i drugu periferijsku opremu), EDITOR, FUN, GRAPHICS, INFO (razni arhivirani priručnici), MISC, PROGRAM (programerske alatke), SYSTEM (sistemski programi), TEST (dijagnostičke alatke, programi za procenu brzine računara itd), UTILITY, VIRUS i YU.

A što se tiče *chat*-a, u pitanju je svet za sebe. Mnogi korisnici *Sezama* su se na ovaj način upoznali i postali dobri prijatelji pre nego što su uopšte imali priliku da se „uživo" sretnu. Neki su postali i više od toga – barem dva para su se upoznala i zbližila preko *Sezamovog chat*-a i privatnih poruka. Neki korisnici (i korisnice) su postali legende *Sezama*; neki su se pitali čemu sve to služi i zašto se drugima toliko sviđa; mnogi su našli na link sa All H... ali šta da vam pričamo, treba doći i sve to isprobati!

Sezam, srećom, ne čine samo konferencije, direktorijumi i *chat* kanali – više od svega toga čine ga ljudi koji nas zovu. Doprinos tih ljudi se, naravno, ne može svesti na brojke, ali *Sezam* ipak automatski održava liste 20NAJ korisnika – oni koji su najviše zvali, ostavili najviše poruka, poslali i primili najviše datoteka, najviše razgovarali... Prostor u „Računarima" nam ne dopušta da objavimo sve te liste, ali ćemo pokušati sa jednim redukovanim spiskom najzanimljivijih kategorija, navodeći tu samo korisnike koji nisu profesionalno vezani sa *Sezamom*. Svima onima sa liste, kao i stotinama onih koji se na njoj nisu našli, zahvaljujemo na velikom doprinosu razvoju našeg SistEma ZA Modemske komunikacije i želimo da i u godinama koje dolaze pošalju i prime još više korisnih i zabavnih informacija i uživaju u novim uslugama koje će *Sezam* ponuditi.

NAJMANJI KVADRAT

Posle par teških, imali smo i jednu prilično jednostavnu pitalicu. I, naravno, primili dosta rešenja, od kojih su mnoga poslata preko Sezama...

Podsetimo se pre svega postavke. Trebalo je pronaći međusobno različite prirodne brojeve A, B, C, D i E takve da je $A^2+B^2=C^2=D^2+E^2$ (zbir kvadrata brojeva A i B jednak je zbiru kvadrata brojeva D i E, odnosno kvadratu broja C). Pri tome se zahtevalo da broj C bude što manji.

Ako pretpostavimo da brojevi A, B, C, D i E neće biti preveliki, do rešenja ćemo lako doći primenom „grube sile". Obzirom da je C kritičan, konstruisaćemo petlju u kojoj on raste, a zatim za svako izabrano C pokušati razlaganje C^2 na dva različita zbira kvadrata. Ako uspemo, našli smo traženo najmanje rešenje.

Jedno od ovakvih rešenja ilustruje program sa slike 1 – njegovom primenom trenutno nalazimo da su brojevi 7, 24, 25, 15 i 20. Očigledno je da rešenja ima beskonačno mnogo i da se jedna grupa njih može dobiti množenjem svakog od brojeva nekom konstantom.

Neračunarski pronađena rešenja uglavnom se zasnivaju na teoremi koju je u svoje vreme dokazao Euklid: ako je (A, B, C) osnovna trojka Pitagorinih brojeva ($C^2=A^2+B^2$), postoje prirodni i međusobno prosti brojevi različite parnosti M i N (M>N) takvi da je A=2MN, $B=M^2-N^2$ i $C=M^2+N^2$.

Za razne M i N dobijamo osnovne Pitagorine trojke (4, 3, 5), (12, 5, 13), (8, 15, 17), (24, 7, 25), (20, 21, 29) itd. Očigledno je da u trojkama koje slede komponenta C mora biti veća od 25 (jer je M>5). Pošto je komponenta C u prve tri trojke prost broj, ne postoji tražena kombinacija u kojoj bi C bilo 5, 13 ili 17. Međutim, za C=25 imamo dve kombinacije: jedna je osnovna Pitagorina trojka (24, 7, 25) a druga trojka izvedena iz prve (4, 3, 5) množenjem sa 5 – dobija je (20, 15, 25) što je i rešenje zadatka.

Prvu nagradu od 2.000 dinara ovoga puta dobija *Aleksandar Kragl* iz Beograda koji je poslao jedino (i veoma dobro obrazloženo) **geometrijsko** rešenje zadatka, na žalost malo predugo da bismo ga ovde objavili. Preostale nagrade smo izvukli – 1.500 dinara dobija *Berislav Todorović* iz Beograda a 1.000 dinara *Josip Maksimović* iz Sarajeva. Nagradu za rešenje dostavljeno modemom (1.500 dinara) dobija *Ivan Sekulović (isekulovic)* iz Novog Beograda. Pohvale za zanimljiva rešenja zaslužili su *Ivan Fačko, Mirza Hadžić, Milovan Kovačević (majk!), Ranko Lazić (rlazic), Nikola Miljković*.

NAGRADNA PREMEŠTALIKA

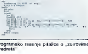
Prirodni brojevi 1-25 raspoređeni su na slučajan način kao na slici 2. U svakom potezu bilo koja dva broja mogu međusobno zameniti mesta, pri čemu je cilj da se brojevi rasporede u rastući niz 1, 2, 3, ... 25. To, naravno, treba uraditi primenjujući što manje poteza.

Zadatak možete rešiti bilo kojim standardnim algoritmom za sortiranje, ali obratite pažnju na to da se, u datom specijalnom slučaju, možda može uštedeti nekoliko zamena.

Rešenja pitalice, zajedno sa **obrazloženim** programima koje ste napisali, pošaljite na uobičajenu adresu „Računari" (za Dejanove pitalice), Bulevar vojvode Mišića 17, Beograd tako da pristignu pre 1. decembra 1991. Sva pisma sa korektnim rešenjima konkurišu za novčane nagrade od 2000, 1500 i 1000 dinara dok će kuponi (ili njihove fotokopije) na koje je upisan identifikacioni broj učestvovati u godišnjem takmičenju rešavača pitalica. Identifikacioni broj dobijate tako što u prve tri kućice upišete **poslednje** tri cifre nečijeg broja telefona, u sledeće dve godinu vašeg rođenja i na kraj dodate dve cifre po izboru. Obratite pažnju da sva rešenja koja šaljete u toku godine nose isti identifikacioni broj. Osim običnom, rešenje ove pitalice možete da pošaljete i elektronskom poštom posredstvom Sezama – konstite menije REDAKCIJA, PITALICE, REŠENJE ili, iz komandnog moda, pošaljite ličnu poruku korisniku pitalice (*mail write pitalice*). Sva rešenja pristigla elektronskom poštom ravnopravno sudeluju u izvlačenju nagrada i konkurišu za **specijalnu nagradu** od 1500 dinara, koja će biti dodeljena najboljem rešenju koje je dostavljeno modemom!

3, 17, 21, 14, 9, 4, 2, 16, 8, 22, 7, 25, 20, 23, 15, 1, 12, 19, 10, 24, 5, 11, 6, 18, 13.

Listing 1



Programsko rešenje pitalice o „zbirovima kvadrata"

70: Brojeve sam sortirao u ______ poteza.

Ime i prezime ______

Adresa ______

Mesto ______

Kompjuter ______ Vreme ______

Identifikacioni broj |_| |_| |_| |_| |_| |_| |_|

iz broja tel. | god. rođ. | po izboru

PC AT 286, 386-SX, 386, 486
ISPORUKA ODMAH

MODEL	SC 2016	SC 2025!!!	SC 3020 SX	SC 3033 CACHE 64 Kb	SC 4033 CACHE 256 Kb
PROCESOR	INTEL 286	INTEL 286	INT 386 SX	INTEL 386	INTEL 486
CHIP SET	U M C	HEADLAND	HEADLAND	U M C	OPTI
RADNI TAKT	16 MHz	25 MHz!!!	20 MHz	33 MHz	33 MHz
LAND TEST	21 MHz	33 MHz!!!	26 MHz	58 MHz	150 MHz
KUĆIŠTE	BABY AT	BABY AT	BABY AT	BABY AT	BABY AT
NAPAJANJE	200 W	200 W	200 W	200 W	200 W
RAM	1 Mb	1 Mb	2 Mb	4 Mb	4 Mb
MAX RAM	5 Mb	8 Mb	8 Mb	16 Mb	32 Mb
FLOPPY	1 2 Mb	1 2 Mb	1 2 Mb	1 2 Mb	1 2 Mb
HARD DISK	ST 157 A 44 Mb	ST 157 A 44 Mb	ST 157 A 44 Mb	ST 157 A 44 Mb	ST 157 A 44 Mb
HD/FD CONT	AT BUS INT 1 1	AT BUS INT 1 1	AT BUS INT 1 1	AT BUS INT 1 1	AT BUS INT 1 1
CORETEST	T/T 1 ms 740 K/S	T/T 1 ms 740 K/S	T/T 1 ms 740 K/S	T/T 1 ms 740 K/S	T/T 1 ms 740 K/S
SER INT	2 RS 232	2 RS 232	2 RS 232	2 RS 232	2 RS 232
PAR INT	2 CENTR	2 CENTR	2 CENTR	2 CENTR	2 CENTR
TASTATURA	101 TAST	101 TAST	101 TAST	101 TAST	101 TAST
GRAF KART	HERCULES	HERCULES	HERCULES	HERCULES	HERCULES
MONITOR 14"	AMBER ili CRNO-BELI	AMBER ili CRNO BELI	AMBER ili CRNO BELI	AMBER ili CRNO BELI	AMBER ili CRNO BELI
CENA DEM	**1.650 DEM**	**1.950 DEM**	**2.450 DEM**	**3.350 DEM**	**4.650 DEM**

18" COLOR MONITOR EIZO 9070S (1200*800) +VGA CARD 1 MB 1024*768	2 300 DEM	*	
14 " VGA COLOR MONITOR +VGA CARD 512 Kb, 1024*768	900 DEM	*	(ZVEZDICOM SU OZNAČENE DOPLATE U ODNOSU NA OSNOVNE KONFIGURACIJE)
HD 105 Mb NEC	550 DEM	*	
HD 130 Mb ST 1144 A 18 ms	650 DEM	*	
HD 210 Mb CONNER 15 ms	1 350 DEM	*	
TOWER KUĆIŠTE 230 W	250 DEM	*	
1 Mb RAM	150 DEM		
IIT 287-10 MHz KOPROCESOR	250 DEM		(100—300% BRŽI OD INTELA 287)
IIT 287-20 MHz KOPROCESOR	300 DEM		(100—300% BRŽI OD INTELA 287)
IIT 387-33 MHz KOPROCESOR	600 DEM		(100—300% BRŽI OD INTELA 387)
FLOPPY DISK 1 44 Mb TEAC	200 DEM		
MIŠ GM 6000 (RS 232)	100 DEM		
STREAMER, 120 Mb	750 DEM		
EPSON LX 400 (A4, 9 PINA)	pozovite nas!		(PRINTERI SU SA UGRAĐENIM YU-ZNACIMA ŠŽČĆĐ I KABLOM ZA VEZIVANJE SA RAČUNAROM)
EPSON LQ 550 (A4, 24 PINA)	pozovite nas!		
EPSON LQ 1010 (A3, 24 PINA)	pozovite nas!		
EPSON LQ 1050 (A3, 24 PINA)	pozovite nas!		
EPSON FX 1050 (A3, 9 PINA)	pozovite nas!		
EPSON LX 1050 (A3, 9 PINA)	pozovite nas!		

UPLATA JE U DINARSKOJ PROTIVVREDNOSTI PARITETA DEM MEĐUBANKARSKOG KURSA NA TRŽIŠTU U BEOGRADU
NAČIN PLAĆANJA: 100% AVANS — ISPORUKA ODMAH
ŽIRO RAČUN 60815-601-64787

PREDUZEĆE „ŠUTLIĆ"
& „MIKRO DIZAJN"

KOSOVSKA 32, / 1. SPRAT, 11000 BEOGRAD
RADNO VREME: 9—17 h, SUBOTOM I NEDELJOM NE RADIMO

TEL./FAX: 011/332-607

GARANCIJA 12 MESECI, SERVIS I ODRŽAVANJE
BEOGRAD 01. 10. '91.

CENE OPREME VAŽE ZA PRAVNA I FIZIČKA LICA (OD 1. 2. '91. PLAĆA SE POREZ 3%)

PRECISE COMPUTERS

SARAJEVO - DOBOJ - TUZLA — 074 / 41-112

RAČUNARI

RAČUNARI	DIN
AT-286-16MHZ	60 050
286-16MHz, 1MB, Baby kućište, FDD 1,2 ili 1,44, ser. + par., 14" mono monitor, YU set	
AT-386SX 20MHz/45MB	98 090
386SX 20MHz, Baby kućište, FDD 1,2 ili 1,44, 45MB HDD, ser. + par., 14" mono monitor, YU set	
AT-386-33MHz/105MB	168 300
386-33MHz, 2MB, 64KB CACHE, Tower kućište, FDD 1,2 ili 1,44, 105MB HDD, ser. + par., 14" mono monitor, YU set	
AT-486-33MHz/426MB	341 630
486-33MHz, 4MB, 64KB CACHE, BIG TOWER kućište, FDD 1,2 ili 1,44, 426MB HDD, ser. + par., 14" VGA mono monitor, YU set	

ŠTAMPAČI

MATRIČNI		DIN
EPSON LX-400	180z/s, 9 pin, A4	27 800
EPSON LX-850	200z/s, 9 pin, A4	57 600
EPSON LX-1050	200z/s, 9 pin, A3	64 800
EPSON FX-850	300z/s, 9 pin, A4	69 600
EPSON FX-1050	300z/s, 9 pin, A3	• 67 500

		DIN
STAR LC-20	180z/s, 9 pin, A4	27 900
STAR LC-200 Color	200z/s, 9 pin, A4	39 800
STAR LC-15	200z/s, 9 pin, A3	45 700
STAR LC-24-10	180z/s, 24 pin, A4	• 39 700
STAR LC-24-15	200z/s, 24 pin, A3	59 300

LASERSKI		
HP LJIIP	4 str/min	189 500
HP LJIII	8 str/min	278 300

PORTABLE		
CANON BJ-10E	80 z/s	34 500
CANON BJ-150E	150 z/s	62 800

CIJENIK Br. 11-2/91 OD 19.11.1991.

Sve cijene su FCO Doboj, Tuzla, Sarajevo
Za većinu artikala isporuka odmah
za ostatak maksimalno 6 radnih dana

NOVELL NETWARE Kompletna paleta
FOTOKOPIR APARAT CANON
TELEFAX (CANON, GUIS, PANASONIC)

PROMPTNA ISPORUKA SVIH DIJELOVA I POTROŠNOG MATERIJALA

HARD DISKOVI		DIN
ST 157A-1	45MB/28ms	• 19 880
ST 351A/X	42MB/28ms	22 920
ST 3096A	89MB/15ms	• 32 900
ST-3144A	120MB/19ms	44 900
ST-1144A	120MB/19ms	39 850
ST-1239A	211MB/15ms	69 370
ST 1480A	426MB/15ms	• 127 300
KALOK SC 3100	105MB/16ms	• 33 700
WD Caviar 280	80MB/16ms	35 230
QUANTUM 210A	185MB/18ms	79 350

DISKETNE JEDINICE		
5.25"	1.2MB	8 900
3.5"	1.44MB	8 700

STREAMER		
Colorado	120MB	33 900
Colorado	250MB	43 850

MEMORIJE I ČIPOVI		DIN
DRAMs	41256 80/70ns	198
DRAMs	44256 80/70ns	• 495
SIMM	256x9 80/70ns	1 600
SIPP	256x9 80/70ns	1 680
SIMM	1Mx9 80/70ns	• 5 300
SIPP	1Mx9 80/70ns	• 5 520
EPROM	27C64	220
EPROM	27C256	398

MREŽNE KARTICE	
Ethernet 8 bita	18 700
Ethernet 16 bita	19 860
Arcnet ANC-115	13 920

KUĆIŠTA		
Baby	+ PS 200W	9 750
Mini Tower	+ PS 200W	12 800
Tower	+ PS 200W	17 500

MATIČNE PLOČE	DIN
AT-286/12MHz	10 980
AT 286/16MHz	12 900
AT-286/20MHz TOPCAT	18 740
386 SX/20MHz	28 560
386 SX/25MHz	29 950
386/25MHz	39 700
386/33MHz + 64K CACHE	• 52 760
386/40MHz + 64K CACHE	62 900
486/33MHz + 64K CACHE	• 89 000

GRAFIČKE KARTICE	
MGP	2 800
MGP YU	3 500
EGA	6 320
VGA (256K, 512K, 1MB) od	6 950

TASTATURE	
XT/AT YU, GR, US 101/102	5 900

KOPROCESORI		DIN
80287-10	AMD	7 600
80C287-20	IIT	14 800
80387-25	INTEL	23 800
80C387-25	IIT	23 500
80C387-33	IIT	• 24 820

KONTROLERI I I/O	
IDE AT-BUS	2 500
IDE AT-BUS + I/O	4 350
AT I/O (S+P+G)	2 890
ADAPTEK ABC2322 ESDI	26 300
WD-7000 FASST2	35 800

MONITORI	
14" Monochrom Dual	12 980
VGA-Mono Datas	15 900
SVGA-Color 1024x768	42 500
SVGA-Color PHILIPS	48 700

INIS-IMPEX

TEL 074/41 112, 41-300, 41 111
FAX 074/41-112

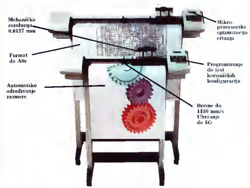


ScanJet
LaserJet
IIIP
III
HEWLETT PACKARD ASORTIMAN
LATINIČNI I ĆIRILIČNI FONTOVI
U BOJI
PUNJENJE TONER KAZETA
CRNO
NOVO!
RECOGNITA
FONT KAZETE
NOVO!
UPS
Kalkulatori
STALCI ZA PRINTERE OD PLEKSIGLASA
PERI HARD®
YUGOSLAVIA
NOVO! HP ScanJet IIc
NOVA ADRESA!
PERIHARD ZAGREB
Klekovačka 28/I, pp 5030, Zagreb
tel 041/263-696, fax 041/263-714
PERIHARD INŽENJERING
Ivana Milutinovića 24, Beograd
tel 011/436-019, 423-383, fax 011/435-513
OPTIK GLAS
STAR LIGHT
STAKLENI
FORTUNA
UP 'N DOWN
PROTECTOR
SUNFLEX
MIKROVLAKNA
FLEX
VEC
Autorizovani dealer
HEWLETT PACKARD
LaserJet IIIp
PROFESIONALNI ZAŠTITNI FILTERI

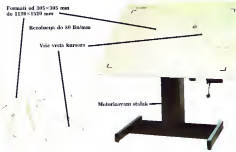

IBC
computer equipment
IBC
TRIESTE (Italy) - Via Caboto 19 - Tel. (040) 82.34.21 - Fax (040) 82.34.25

Summagraphics™ HOUSTON INSTRUMENT

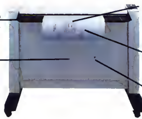
Ploteri



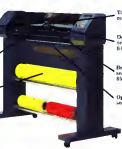
Digitalizatori



Skeneri



Sekači



Vrhunske grafičke periferije, softver i inženjering

NOVO! NOVO! FAMILIJA PLOTERA EDMP 160 sa još boljim performansama

kma D.O.O.

11070 N. Beograd
Milentija Popovica 9 (Sava centar)

Tel. 011/222-19-28
Tel. 011/439-618
Tel. 011/435-715
Tel. 011/222-43-22, lokal 453
Fax 011/455-785, 605-578

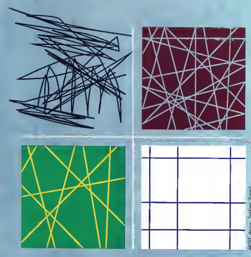
Ovlašćene prodajne organizacije

SoftWell d.o.o.
Kompjuterski inženjering
Pozorišni trg 1
21000 Novi Sad
Tel. 021/51-999, 52-396

EI „Računari" — Niš
Bul. V. Vlahovića 80—84
18000 Niš
Tel. 018/325 461, 55 583
Fax 325 726

Lanaco
Braća Pavlića H
78000 Banjaluka
Tel. 078/33-992, 46 717
Fax 33-992

Makpetrol kompjuteri
Kompjuterski inženjering
Ul. „Orce Nikolov" bb
91000 Skopje
Tel. 091/229-676, 221 040
Fax 229 676



MOST design, Srećko Vujović

The Art of Networking

Računare treba povezati u mreže da bi se optimalno iskoristili ljudski i mašinski resursi To je novi korak u razvoju računarstva i informatike Vreme individualnih i velikih mainfreme računera u poslovnom okruženju je prošlo

Da bi se formirala mreža nije dovoljno samo kupiti kutije Dođite kod nas sa znanjem o zahtevima koje sistem treba da ispuni u praksi i izradićemo projekat kompletnog informacionog sistema prema Vašim potrebama Naši stručnjaci su kompetentni konsultanti koji mogu rešiti Vaše probleme u oblasti poslovne obrade Poslovni softver koji vam nudimo predstavlja vrhunska dostignuća u ovoj oblasti

Odabrali smo „Novell", najvećeg proizvođača softvera za lokalne računerske mreže, jer se „Most" pouzdaje samo u proizvođače koji garantuju kvalitet Sa „Novell" NetWare operativnim sistemima naše mreže ispunjavaju sve Vaše zahteve za povezivanje PC mini i mainframe računara Raspolažemo znanjem i iskustvom koje Vam garantuje da će Vaša mreža raditi uz maksimalno iskorišćenje performansi Pored toga kod nas ćete dobiti kompletnu stalnu podršku, otklanjanje svih kvarova u redu mreže u roku od 24 časa kao i telefonsku podršku

Kupovinom u „Mostu" tako dobijate kompletno rešenje Vašeg problema Naši sistemi su projektovani tako da budu tehnički fleksibilni i finansijski povoljni Obratite nam se zahtevom za ponudu za Vaš konkretan problem ili zahtevajte listu naših referenci

RADNO VREME OD 9 – 17 ČASOVA



Zašto bi vaš navi kompjuter trebalo da bude "M O S T"

Kod nas se računari ne prodaju po svaku cenu – prodajemo samo najbolje. Ovde ćete dobiti savet, pomoć u radu ili projekat za kompleten informacioni sistem. Kompjuteri i informatika su naš životni poziv, pa možete biti sigurni da ćete kod nas uvek dobiti kompetentnu stručnu pomoć.

Ciljevi kompanije „Most" uvek su bili jasni: stvoriti bazu zadovoljnih korisnika, dostići i održavati visok nivo kvaliteta, postati najbolji. Mislimo da su ti ciljevi ostvareni, a vi ste tome doprineli. Danas biti najbolji određuje koje će kompanije preživeti. „Most" će još dugo biti sa vama.

Nazovite telefonom – možda tako rešimo vaš problem.

Naš oglas više nije običan cenovnik. Mi se razlikujemo od drugih. Trudimo da sjedinimo umetnost i tehnologiju.

JUNIOR 286/16 • ROSE: 386SX/20 • ASSISTANT: 386/33 • GENIUS: 386/40
DOLPHIN: 486SX/20 • STAR: 486/25 • GOLIATH: 486/33

NOVO: NetWare-Ready diskovi – tvrdi diskovi spremni za instalaciju »Novell« mreža sa obavljenom analizom površine diska

PREDUZEĆE »MOST«, TEL. 422-170, FAX. 420-141, BEOGRAD, KRALJA BODINA 2

MOST


EPSON
VELIKO
IME,
POUZDAN
KVALITET!
EPSON
EPSON
EPSON
COMPUTERS



Da je imao domaću verziju WordPerfecta 5.1, možda bi danas struja bila besplatna

Tesla bi bio oduševljen! Najmoćniji i najpopularniji tekst procesor u svijetu - u potpunosti je preveden i prilagođen domaćem korisniku! Članke, bilješke, rasprave i knjige mogao bi pisati pomoću vrhunskog softverskog alata - WordPerfecta 5.1, koristeći njegove neograničene mogućnosti. Tabele, grafika, stupci, formatiranje, kontrola štampanje, redefiniranje štampača. I za Teslu možda najvažnije - mogućnost jednostavnog kreiranja kompleksnih matematičkih formula i jednadžbi, dali bi ovom misliocu alat dostojan njegovog lucidnog uma. Nekoliko poteza mišem - klik, klik - i gotov je članak s teorijskim osnovama novih, svemirskih izvora energije, besplatnih i izobilnih. Teško da bi se čak i skeptični financijeri mogli oduprijeti ovako jasno prezentiranim zamislima!

No domaća verzija WordPerfecta pristupačna je svima, a ne samo genijima poput Tesle. Prvi puta u povijesti domaćeg računarskog tržišta, jedan od ključnih svjetskih softverskih alata u potpunosti je lokaliziran, u suradnji WordPerfect Corp. i generalnog distributera za Jugoslaviju, Perpetuum d.o.o. Sve komande, sve pomoćne datoteke, pa i veliki dio dokumentacije preveden su kako bi se domaćim korisnicima pružila prilika da maksimalno iskoriste neograničene mogućnosti ovog alata.

WordPerfect 5.1 - domaća verzija. Sad kad je tu, zamislite mogućnosti!

OVLAŠTENI DISTRIBUTER

SoftWell

KOMPJUTERSKI INŽENJERING

T. MARKOVIĆA 14 21000 NOVI SAD TEL. 021/51-999 52 396 FAX 51 999

GENERALNI DISTRIBUTER

perpetuum d.o.o. Zagreb

USKORO WINDOWS VERZIJA!



SCO

THE SANTA CRUZ OPERATION

Softverska kompanija 1990. godine

Operativna okruženja

SCO System V

Porodica SCO Sistema V obuhvata SCO UNIX System V/386 Release 3.2 i SCo XENIX System V višekorisnički multitasking operativni sistemi za mikroračunare bazirani na industrijskim standardima

SCO UIX System V/386 Release 3.2

Robustna, potpuno licencna verzija AT&T UNIX Sistema V sa dodatnim poboljšanjima koja je isključivo ugradio SCO

SCO XENIX

Industrijski standardan UNIX sistem za personalne računare bazirane na intelu

SCO MPX

Multiprocesorsko proširenje za računare bazirane na 386 i 486 koji rade pod SCO UNIX Systemom V/386

SCO VP/ix

Pod SCO VP/ix višestruke MS DOS aplikacije mogu ići paralelno sa konvencionalnim XENIX i UNIX sistemima

Komunikacije i mreže

SCO TCP/IP

Povezivanje preko mreže između raznih operativnih sistema i računara

SCO NFS

Networked File i Data Sharing za SCO UNIX i Open Desktop

SCO XENIX-NET

Microsoft mreže za XENIX sisteme. Lokalna mreža namenjena da specifično poveže personalne računare koji rade pod XENIX-om i DOS-om

SCO UniPATH SNA-3270

Višekorisnička veza između mikro i glavnog računara — ulaz višekorisničkih personalaca u IBM SNA mreže

Radne stanice

Open Deskop

Potpuni grafički operativni sistem. Open Desktop vam omogućava da koristite moćne grafičke aplikacije u prozorima, da manipulišete podacima, pristupate datotekama i aplikacijama na umreženim sistemima i da koristite DOS i UNIX sisteme

Poslovna automatizacija

Microsoft Word 5.0

Višekorisnički tekst procesor za UNIX sisteme

SCO Portfolio

User friendly interfejs za SCO System V uključujući aparate za automatizaciju kancelarijskog poslovanja, kao što su elektronska pošta (e-mail) i kalendar

SCO Professional

Lotus 1-2-3 workalike za SCO System V

SCO FoxBASE+

dBASE III plus workalike za SCO System V

Svetski lider u rešenjima za softver otvorenih sistema

MAC RITA PC COMPUTING

AWARDS

SCO je dobio priznanje kao Softverska Kompanija Godine na ceremoniji dodele RITA nagrada



DISTRIBUTER ZA JUGOSLAVIJU

ENERGOPROJEKT

HOLDING KORPORACIJA

MDD ENERGODATA

11070 Beograd, Lenjinov bulevar 12, p. fah 20
Tel (011) 138-520, 139-712, 222-3072
Fah (011) 222-4780 • Tlx ENDATA YU 11764